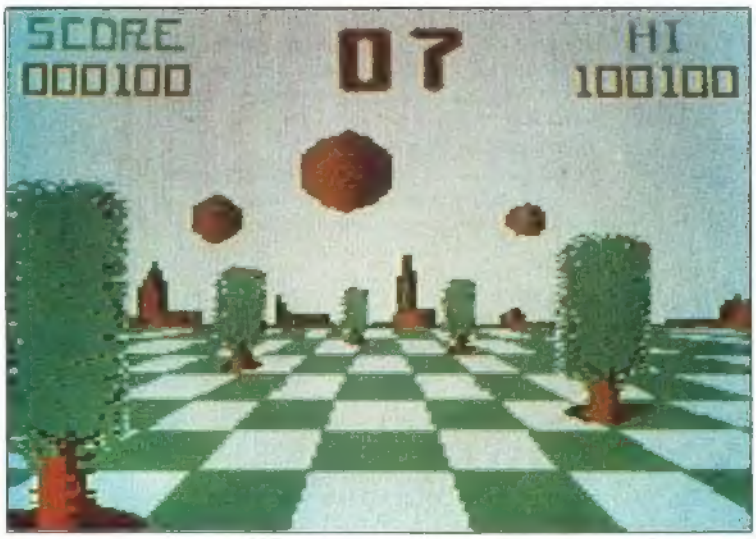
Un "megagame" (per il 64) tutto nostro



di Marco Pesce

Certe volte si ha in mente di fare una cosa e poi... È proprio il mio caso: l'idea di questa rubrica è nata dall'idea di un'altra rubrica. Avevo intenzione di dedicare questo spazio a una sorta di «Scuola di Videogame» e l'iniziativa sembrava interessante. Arrivo al momento di cominciare, ma qualcosa mi blocca. Probabilmente sarà stata la voce della coscienza, anzi, ancora più probabilmente è stato un lampo di genio.

In Italia non sappiamo fare videogiochi.. FALSO!!!

In Italia non vale la pena fare videogiochi... VERO!!!

Pirati e abbondanza straniera fanno sì che un'attività del genere conduca a dei guadagni estremamenti bassi... anzi è già una fortuna se c'è un guadagno. Quello che non posso digerire è la nomina di incapace, quindi al limite basterebbe anche la gloria. Ma di quale gloria sto parlando?... «Un programma con le scritte in italiano è sicuramente pirata». Bella frase; estremamente incoraggiante. È da questa frustazione che nasce la grande pensata: «Perché non unire le forze disperse e sconfiggere il nemico?». Sono sicuro che i più svegli di voi avranno già capito. In Italia nessuno si è mai dedicato seriamente alla programmazione di videogiochi e questo lo abbiamo già detto, ma la cosa più interessante è che per questo motivo molti «solitari» hanno sviluppato conoscenze nel campo con «mezzi propri», con tutte le dovute conseguenze. Qualcosa mi suggerisce che fra di voi ci sono tantissimi (o almeno numerosi) graficomputeromani, musicomputeromani, videocomputeromani e via dicendo e un ulteriore suggerimento mi fa concretizzare l'ipotesi che ognuno di voi ha come grande aspirazione quella di realizzare un videogioco fantastico. A questo punto abbandoniamo i preamboli ed entriamo nel vivo della questione.

Ragazzi, si può fare!

Insieme realizzeremo uno tra i più bei videogiochi della storia (ooohhh!! Forse esagero, ma in parte dipenderà anche da voi). Tutti sono autorizzati a partecipare... proprio tutti; anche se amate di più divertirvi che pensare al divertimento degli altri (traduci: anche se passate tutto il vostro tempo libero incollati al joystick) questa è la vostra occasione, anzi forse il maggiore aiuto verrà proprio dai videogiocatori incalliti. Con il contributo di voi lettori giungeremo alla realizzazione di un qualcosa di estremamente interessante. Una precisazione: bisogna essere gente volenterosa e paziente; nulla di grande si realizza in cinque minuti. Bisognava scegliere un computer e con estrema tranquillità si è scelto un Commodore 64... è il più adatto (da sottintendere: «ce l'ha un sacco di gente»).

L'impresa è ardua e naturalmente non mancheranno i problemi (... e quando mancano), ma con la calma...

Avete notato l'abuso di «puntini» che faccio; è l'ansia della presentazione, ma è anche una mia mania.

Certamente c'è qualcuno che non ha ancora le idee molto chiare ed è pienamente giustificato. Qui si parla di megagiochi quando a mala pena si riesce ad entrare nella sufficienza. Ci vuole un'organizzazione efficace e state tranquilli che si farà tutto il possibile per ottenerla.

Non ho ancora chiarito quale sarà il sistema di comunicazione con i lettori: sarà forse tra le parti più complicate del progetto, lo ripeto ancora, occorre comprensione. Il problema non sta tanto nel mezzo di comunicazione (ovviamente il vostro materiale sarà inviato qui da noi per posta), ma nei tempi. Mi spiego: come molti di voi intuiranno, occorre un certo periodo per realizzare MC, di conseguenza voi state leggendo un articolo scritto a fine ottobre. Quando comincerà ad arrivare il vostro materiale, avremo uno sfasamento di due mesi tra quello che voi leggete e quello che arriva qui in redazione.

Parliamo ora di quello che dovrebbe essere «il vostro materiale».

Anche se non siete programmatori e desiderate partecipare potete farlo.

Ci occorrono anche i suggerimenti.

Cominciamo ad organizzarci. Bisogna innanzi tutto partire da un'idea di base, dalla quale ovviamente si svilupperà il gioco. E qui ci sarà da sbizzarrirsi: ognuno può pensare a un particolare soggetto al quale è interessato e mandarcelo (l'idea, non il gioco fatto), quindi quello che per ora dovete fare è descrivere un gioco così come lo vorreste, anche approssimativamente, poi in base alle vostre preferenze, si deciderà quale sarà la via migliore da prendere. Ovviamente il vostro compito non si limiterà a questo; una volta definito in grandi linee, il gioco necessiterà di molte altre cose oltre alla programmazione vera e propria e di questo potremo rendercene conto nelle prossime puntate. «Ma qualcuno dovrà pur programmare»... già... il sottoscritto non ha nessuna intenzione (o quasi) di caricarsi di tutto il lavoro, quindi preparatevi anche per questo, anzi ricordate che all'inizio dell'articolo accennavo ad una scuola di videogiochi? Penso proprio che sarà di aiuto a un bel po' di gente, che se non altro comincerà a capirci qualcosa anche dell'aspetto creativo vero e proprio, quindi ogni puntata ci dedicheremo anche alla «scuola». Niente di noioso, questo posso

Foto 1 - Nella foto di apertura la schermata del Simil Space Harrier.

garantirvelo, ma tuttavia realmente utile anche per quelli un po' più avanzati.

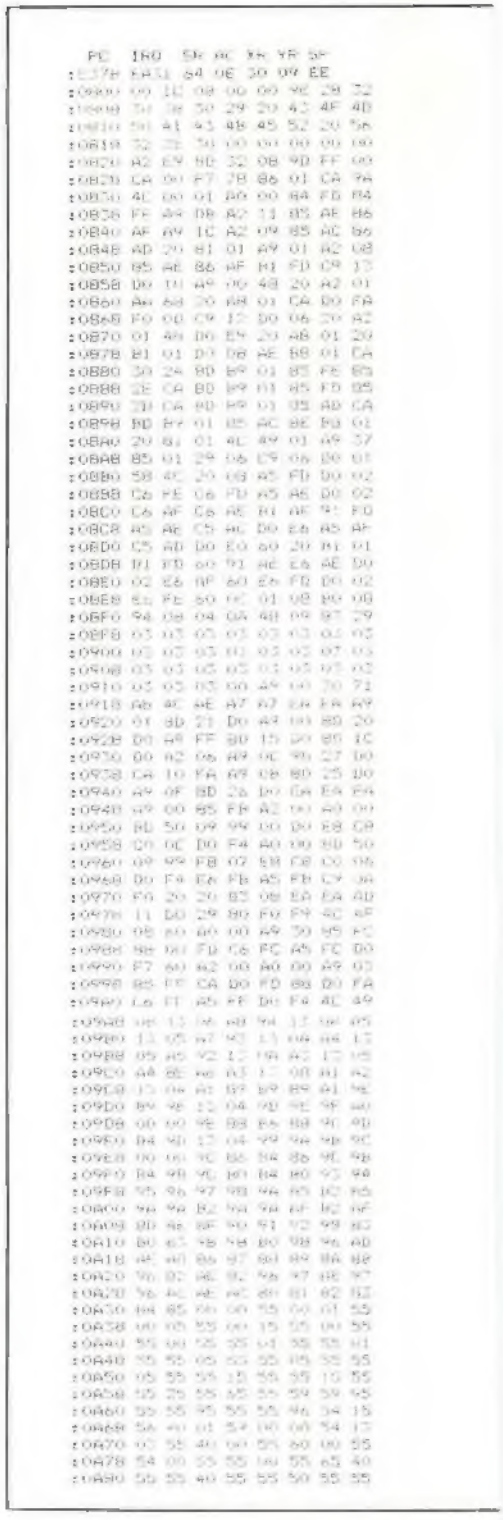
L'occhio vuole la sua parte e quindi ho ben pensato di fornire l'articolo di qualche foto, che hanno il duplice scopo di attirare la vostra attenzione a questa rubrica e di farvi immaginare il livello al quale deve essere realizzato il videogame. Visto che roba! Sono in parte prese da giochi esistenti in commercio, che certamente conoscerete, ma alcune (come la 1 la 2 o la 3) sono di mia realizzazione. Attenzione, sono solo dei modellini non funzionanti, ma volendo... capito ragazzi! Datevi da fare con le idee, tirate fuori qualcosa di nuovo, di originale, che qui ci impegnamo seriamente!

Se vi è venuto in mente di pormi la domanda «ma se sei così bravo perché non te lo fai da solo il MEGAGAME?» allora non avete capito niente e non avete la minima idea di quanto sia noioso procedere in solitudine.

Ed ora una piccola precisazione. Io trovo l'iniziativa divertente e spero siate d'accordo con me, quindi è ingiusto pensare che, una volta finito, il gioco sarà messo in commercio, anche perché difficilmente ci sarebbe un profitto (e il motivo lo sappiamo tutti). Inoltre (sono sicuro che la vostra partecipazione non mancherà) tutti i collaboratori validi saranno nominati sia nella presentazione del gioco che nelle pagine di MC, quindi anche se i cracker sono in agguato (ma sarà molto dura per loro), la gloria, almeno quella, non mancherà.

Cominciate a pensare... pensate bene e... sbrigatevi a mandarci le idee.

Nel frattempo parliamo della...



Scuola di videogame

Qualcuno di voi avrà certamente pensato (speriamo) che l'idea è valida per cominciare alla grande nel mondo dei videogiochi, ma se si vuole anche continuare occorre un minimo di preparazione. È ovvio che c'è chi questa preparazione presume di averla già acquisita abbastanza, ma come si dice: «Non si finisce mai di...» (frase da completare come compito a casa).

Con quest'altra iniziativa si cercherà di rendere chiari gli aspetti più oscuri e cioè anche qui il lettore potrà intervenire (spero di poter dedicare un po' di spazio alla posta di questa rubrica), chiedendo consigli e chiarimenti, o anche suggerendo consigli e proponendo argomenti. Per i primi tempi dovrete subirvi i miei deliri, poi ci sarà un effettivo esame delle vostre esigenze. C'è da chiarire comunque che non tratteremo solo di programmazione e di linguaggio macchina, ma altri argomenti saranno, ad esempio, la grafica, il sonoro, gli sguardi alla concorrenza ecc. ecc. Tanto per cominciare vorrei occuparmi proprio di quest'ultimo, assicurando al collega Francesco Carlà, che non ho intenzione alcuna di rubare il suo posto, in quanto lui tratta i giochi dal punto di vista del giocatore, noi invece li osserveremo dal punto di vista del realizzatore.

Sto sbirciando nella mia lista di programmi per scegliere qualcosa di interessante da esaminare. La mia ricerca termina quando leggo un nome che dovrebbe essere noto ai più; si tratta di «SPACE HARRIER». Non è proprio nuovissimo, ma la prima volta che lo vidi in sala giochi rimasi sconvolto dalla sua bellezza (30 giorni al S. Giovanni). Un mio amico mi fece notare che di esso se ne sarebbe vista una versione anche per home computer e io da esperto programmatore sapevo che, per quanto il collega scelto per la conversione poteva essere in gamba, i miracoli non era certo in grado di farli. Per chi non lo conoscesse (ma se ne sono dette davvero tante), si tratta di un cosiddetto «spara e fuggi», con grafica tridimensionale in prima persona, molto ben definita, e sonoro digitalizzato (per la grafica posso dirvi che somiglia a quella della foto 1). Il gioco consiste nel guidare un personaggio armato di razzo sputafuoco, che può volare in diverse direzioni evitando di farlo scontrare con missili e oggetti del paesaggio. Da questo è facile intuire che se non fosse per la parte tecnica, rimarrebbe ben poco del suo fascino.

La trasposizione per C64 però, deluse molte aspettative (ve l'avevo detto io), in quanto ovviamente le tecniche realizzatrici del COIN-OP non erano molto adatte alle capacità del Commodore. Ad esempio, gli oggetti del paesaggio entrano in scena dall'orizzonte per poi ingrandirsi sempre più. Tale effetto grafico deve dare la sensazione di movimento, e a dire il vero è resa molto bene, in quanto tutto si svolge con esagerata fluidità e in aggiunta a questo c'è un pavimento quadrettato (continuate ad osservare la figura 1) che fa da sostegno per gli oggetti (e il terreno) e quindi si sposta con essi (con quelli che sono piantati in terra). L'ingrandimento degli oggetti, molto ma molto probabilmente, è stato realizzato con la tecnica che mi appresto a descrivervi. Nella macchina sono memorizzate tutte le figure alla massima grandezza. Quando un oggetto entra in scena dall'orizzonte, la sua immagine sullo schermo è molto piccola e a far questo è dedicata una routine che ha la facoltà di stampare sul quadro l'immagine sorgente a varie grandezze. Un'immagine più piccola non si ottiene stringendo i pixel, ma catturando alcuni pixel della figura originale e stampandoli uno accanto all'altro. Se la risoluzione della macchina è molto elevata, sembrerà, a vista d'oc-

```
:0A80 50 55 55 54 55 55 54 55
:0A90 55 56 55 55 55 95 56 12
:0A98 55 08 95 55 95 55 55 56
:0AA0 A9 58 40 2A 65 00 00 A5
:0AA8 00 00 A5 00 58 00 00 58
:0AB0 00 00 54 00 0C 54 C0 00
:0AB8 3F 30 00 5F 00 01 5F D0
:0AC0 00 5C 00 00 5C FF C0 6C
:0AC8 55 B0 BF 95 6C FF AA 5B
:0AD0 BF A6 A7 BE 9B EB B1 6C
:0AD8 3F B0 BF BC BC 2A FF BC
:0AE0 3F FB 3C 03 31 3F FC 2B
:0AE8 3B 00 3F 13 03 15 00 00
:0AF0 75 00 00 15 00 01 F5 00
:0AF8 00 C5 00 01 F5 00 C7 F5
:0B00 00 00 FD 00 A6 C0 0E 55
:0B08 7B 35 AB F0 E5 A0 F0 3F
:0B10 9A F0 FB EC 3D 3B 3C FF
:0B18 7E CF 0F FF AB 3C EF FC
:0B20 0E E0 C0 AE E0 30 3E F0
:0B28 00 3F 00 0F 00 3D 0B C0
:0B30 3B 0D C0 2C 0B 00 2C 0B
:0B38 C0 2E 0B F0 25 0B C0 2A
:0B40 02 F0 0F 02 BC 37 02 B0
:0B48 C0 00 BC 0D 00 0L 15 00
:0B50 AF 2A 00 1F 0B 00 17 C2
:0B58 00 07 C2 00 07 EF 00 04
:0B60 F8 00 01 AC 00 01 AB 00
:0B68 00 5A 00 20 00 FF 2C 00
:0B70 FC 38 00 3C 38 00 FC 08
:0B78 00 FC 58 00 FC AC 03 FC
:0B80 F0 00 01 CC 0F F0 0C 03
:0B88 A0 9C 0F A0 55 3F C0 AA
:0B90 3E B0 3B FF B0 A0 FA 00
:0B98 A3 FE 00 BF AB 00 BE EB
:0BA0 00 0F A0 00 3A E0 00 EE
:0BA8 B0 13 04 05 00 00 55 00
:0BB0 01 55 00 05 55 00 15 55
:0BB8 00 55 55 00 55 55 01 55
:0BC0 55 01 55 55 05 55 55 05
:0BC8 55 55 15 55 55 15 55 56
:0BD0 25 55 A5 95 59 59 95 55
:0BD8 55 95 55 56 A6 54 15 56
:0BE0 40 01 59 00 00 58 13 03
:0BE8 50 00 00 55 00 00 55 80
:0BF0 00 55 50 00 55 54 00 55
:0BF8 95 00 55 55 00 55 55 40
:0C00 55 55 40 35 55 50 55 55
:0C08 50 55 55 50 55 59 50 55
:0C10 55 50 55 55 50 56 55 50
:0C18 55 55 50 5A A5 A0 00 A9
:0C20 90 00 02 90 00 00 50 00
:0C28 58 00 00 54 00 0C 54 C0
:0C30 00 3F 30 00 3F 00 00 3F
:0C38 C0 00 3C 00 00 3C FF C0
:0C40 AE 5A A0 FF AA 5B BF A6
:0C48 A4 BE AB EB B3 0C 0C B0
:0C50 BF BC BC 2A FF BC 3F FB
:0C58 3C 03 3B 3F 0C 2B 3B 00
:0C60 3 0B C0 3B 0B C0 2C 13
:0C68 03 D0 00 00 50 00 07 D0
:0C70 A0 03 10 00 05 D0 00 0F
:0C78 D0 00 03 F0 02 AB 30 39
:0C80 35 F0 9A AB F0 FA AB F0
:0C88 E3 B2 F0 E0 CB F0 FB 7C
:0C90 30 BE AC F0 BF F0 30 B7
:0C98 00 30 B0 C0 F0 C0 00 F0
:0CA0 B0 03 F0 B0 00 F0 00 02
:0CA8 C0 0B 02 F0 09 02 BC 09
:0CB0 02 B0 0E 00 BC 03 00 AF
:0CB8 0C 00 AC 30 00 2F 03 00
:0CC0 1B CA 00 07 C2 00 05 F0
:0CC8 00 01 F0 00 01 FB 00 02
:0CD0 BE 00 00 A8 00 03 56 00
:0CD8 00 16 00 00 09 13 06 30
:0CE0 00 FC D8 00 FC 58 00 FC
:0CE8 AC 03 FC F0 00 EC CC 0F
:0CF0 F0 0C 03 A0 9C 0F A0 AA
:0CF8 3E B0 3B FF B0 A0 FA 00
:0D00 A3 FE 00 BF AB 00 BE E3
:0D08 00 0F A0 00 3A E0 00 EE
:0D10 B0 00 A5 40 17 4B 15 01
:0D18 01 55 00 05 55 00 15 55
:0D20 00 15 55 00 55 55 01 55
```

```
:0D28 55 01 55 55 05 55 55 05
:0D30 55 55 15 55 55 15 55 59
:0D38 25 55 95 55 65 65 95 55
:0D40 59 96 50 55 56 40 04 59
:0D48 00 00 56 00 00 58 00 01
:0D50 5A 00 20 00 A0 00 00 54
:0D58 00 00 54 00 00 55 00 00
:0D60 55 40 00 55 50 00 55 54
:0D68 00 55 54 00 55 55 00 55
:0D70 55 00 55 55 00 55 55 00
:0D78 55 55 00 55 55 00 55 55
:0D80 09 AA 56 00 02 99 06 00
:0D88 29 00 00 05 00 00 0D 00
:0D90 00 05 00 00 5F 00 00 5F
:0D98 C0 00 5C 00 00 50 FF 00
:0DA0 AE 5A C0 FF A9 AE BF 9A
:0DA8 93 BE AF AF B3 33 73 B0
:0DB0 BE 33 BC 3F EE 3C 0C EE
:0DB8 3F 30 AE 3B 00 FF 0B C0
:0DC0 E2 0B C0 B3 0B 00 B3 0B
:0DC8 C0 9B 0A E0 93 02 E0 2F
:0DD0 02 B0 CC 00 00 31 00 00
:0DD8 FB 00 00 3F 00 0A B3 00
:0DE0 E5 5F 00 AA BF 00 EA BF
:0DE8 00 BF 2F 00 B0 AF 00 EF
:0DF0 C3 00 FF 03 00 0C 03 00
:0DF8 00 0F 00 00 0F 00 C0 3F
:0E00 00 00 0F 00 BC 3F 00 B0
:0E08 2F 00 B0 3F 00 00 2B 00
:0E10 C7 FC 13 03 7C AA 00 2C
:0E18 25 00 2F 0A 00 1F 0A 00
:0E20 0F B0 00 0A B0 00 01 A5
:0E28 00 01 55 00 00 94 13 25
:0E30 A5 A0 00 0F E0 00 0E B0
:0E38 00 3F B0 00 FA 00 00 F8
:0E40 00 00 AC 00 00 E0 00 00
:0E48 40 13 29 55 00 05 55 00
:0E50 15 55 00 55 55 02 55 55
:0E58 01 55 55 01 55 55 05 55
:0E60 55 05 55 55 15 51 55 15
:0E68 55 65 95 65 95 95 55 55
:0E70 95 55 65 55 00 10 58 00
:0E78 00 58 00 00 58 00 00 54
:0E80 00 C0 5C C0 00 5C 13 06
:0E88 40 00 00 50 00 00 54 00
:0E90 00 A6 00 00 55 00 00 55
:0E98 40 00 55 50 00 55 50 00
:0EA0 55 50 00 55 50 00 55 50
:0EA8 00 95 50 00 55 50 00 29
:0EB0 90 00 02 90 00 00 50 00
:0EB8 00 00 00 00 50 00 03 10
:0EC0 00 03 D0 00 00 5F FC 00
:0EC8 A0 5D 03 FE A5 B9 BE 9A
:0ED0 AF BE BE BE B2 FB CF BC
:0ED8 AF FB 3C FF BE 3C 32 E8
:0EE0 38 03 FC 08 03 BB 0B 02
:0EE8 CE 00 02 CE 08 C2 56 0B
:0EF0 03 AB 02 C0 FC 02 CC 03
:0EF8 02 F0 E7 01 B2 AA 00 20
:0F00 BE 00 5C 28 00 AB 30 00
:0F08 55 F0 00 AB F0 00 9B F0
:0F10 00 EE F0 00 CC 30 00 AB
:0F18 F0 00 FC 30 00 30 F0 00
:0F20 00 F0 00 03 F0 00 00 F0
:0F28 00 03 F0 00 03 F0 00 03
:0F30 F0 00 03 B0 00 02 B0 00
:0F38 0F B0 00 3E 00 00 FE 00
:0F40 00 FB 13 08 2F AF 00 0A
:0F48 C3 00 05 BE 00 02 A5 15
:0F50 34 E3 00 00 A0 00 00 B0
:0F58 00 00 40 13 37 01 50 00
:0F60 05 55 00 05 55 00 55 55
:0F68 01 55 55 09 55 55 05 55
:0F70 55 05 55 55 15 55 55 25
:0F78 59 55 55 95 55 95 55 95
:0F80 99 45 5A 54 00 00 50 00
:0F88 00 50 00 00 53 00 00 5C
:0F90 00 00 55 00 00 50 00 00
:0F98 69 50 09 13 07 40 00 00
:0FA0 60 00 03 50 00 00 54 00
:0FA8 00 56 00 00 55 00 00 55
:0FB0 00 00 55 00 00 55 00 00
:0FB8 55 00 00 96 00 00 09 00
```

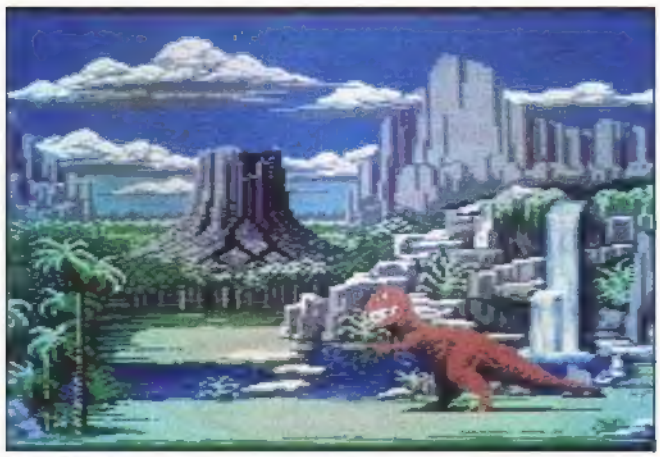
```
:0FC0 00 95 00 00 00 00 00 0D
:0FC8 00 00 01 00 00 3D 00 00
:0FD0 0F 00 00 5F 13 03 2F 9A
:0FD8 4E 2F 31 CC 2C BC CE 2C
:0FE0 3F BB 0C 0F BB 0F 03 FC
:0FE8 0F C3 BB 0F C2 C3 03 C2
:0FF0 74 02 C2 54 02 C0 FC 02
:0FF8 B3 30 00 B0 E4 00 75 AA
:1000 00 3C 8C 00 1C 2B 00 0E
:1008 EF 00 06 C3 00 02 BE 00
:1010 02 6A 13 04 AB C0 00 08
:1018 C0 00 FC C0 00 F0 C0 00
:1020 C0 C0 00 03 C0 00 02 C0
:1028 00 03 C0 00 03 C0 00 03
:1030 C0 00 02 C0 00 3F 00 00
:1038 3E 00 00 38 00 00 FC 00
:1040 00 F8 00 00 A0 00 00 B0
:1048 00 00 40 13 04 05 00 00
:1050 15 50 00 55 54 01 55 55
:1058 09 55 55 05 55 55 25 55
:1060 55 15 55 55 25 65 55 95
:1068 55 65 99 15 69 54 00 02
:1070 50 00 00 50 00 00 50 00
:1078 03 5C 00 03 50 00 03 69
:1080 70 05 FE 9F 6B BE AC EB
:1088 BC 1C 63 13 10 40 00 00
:1090 40 00 00 40 00 00 40 00
:1098 00 40 00 00 80 00 00 40
:10A0 00 00 40 00 00 40 00 00
:10A8 40 00 00 40 00 00 C0 00
:10B0 00 C0 00 00 C0 00 00 C0
:10B8 00 00 C0 13 03 30 F3 30
:10C0 3C 3F 00 0C 30 C3 0C 20
:10C8 B0 0C 28 B3 08 3A C3 08
:10D0 3F CF 03 06 CE 02 EA BE
:10D8 01 CC BE 00 CA 3C 00 BE
:10E0 F8 00 6C B0 00 2A 50 15
:10E8 16 C0 00 00 C0 00 00 C0
:10F0 00 00 C0 00 00 C0 00 00
:10F8 C0 13 31 14 00 00 55 80
:1100 09 55 90 05 55 54 15 55
:1108 54 15 55 54 56 65 54 95
:1110 55 54 60 00 08 60 00 04
:1118 5C 00 0C 70 00 3C 7F C0
:1120 A0 FA 6A AC BA EC BC B2
:1128 FB BC 30 28 00 20 28 0C
:1130 2C 3C 0C 2C 14 0C 0C 3C
:1138 08 00 1C AB 80 02 2F B0
:1140 0B B4 00 01 AB 13 36 14
:1148 00 01 55 00 05 55 40 15
:1150 55 50 15 65 90 95 56 50
:1158 54 10 80 80 00 30 70 00
:1160 00 70 00 30 FA 7A B0 B0
:1168 A0 B0 B3 FF 00 20 F0 30
:1170 20 A8 30 2C EC F0 08 0C
:1178 E0 03 0B C0 03 AE 00 01
:1180 BF 00 00 BC 13 06 40 00
:1188 05 54 00 15 55 00 15 95
:1190 00 95 55 00 40 02 00 40
:1198 B3 00 40 03 00 F9 2B 00
:11A0 BC 2C 00 3E C3 00 32 07
:11A8 00 20 C2 00 02 BE 00 04
:11B0 BC 00 02 20 13 11 01 00
:11B8 00 15 50 00 15 50 00 91
:11C0 90 00 40 00 00 7C B0 00
:11C8 BC 30 00 03 00 00 31 30
:11D0 00 11 30 00 0E C0 13 10
:11D8 55 55 55 00 00 00 00 00
:11E0 00 00 00 00 00 00 00 00
:11E8 00 00 00 00 00 00 00 00
:11F0 00 00 00 00 00 00 00 00
:11F8 00 00 00 00 00 00 00 00
:1200 00 00 00 00 00 00 00 00
:1208 00 00 00 00 00 00 00 00
```

Programma che realizza una sequenza di ingrandimento interamente con sprite.

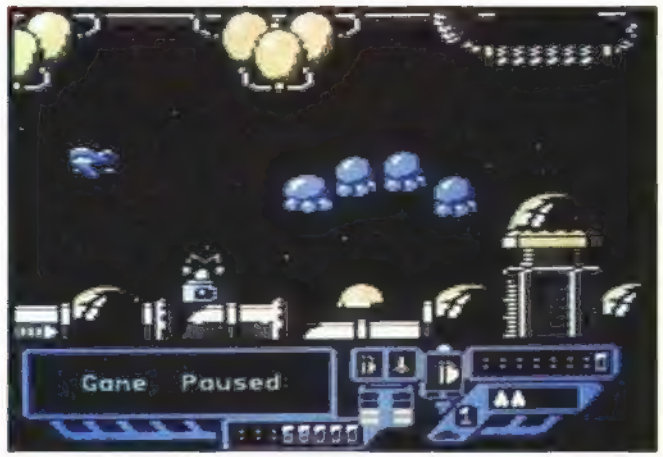
chio, che c'è un effettivo ingrandimento dell'oggetto e non ci si renderà conto che alcuni particolari (omessi in lontananza) si evidenziano solo verso la massima grandezza. Questo sistema pecca in un solo punto; non è estremamente rapido, in quanto ad ogni minimo ingrandimento occorre ricalcolare tutta l'immagine e quindi tutti i pixel da prendere dal sorgente per la stampa, ma il COIN-OP della sala è dotato di una scheda molto veloce e non ci sono di questi problemi. Purtroppo il beneamato C64 non è altrettanto rapido, quindi deve accontentarsi di una tecnica più dispendiosa, proveniente dalle tasche della memoria; ogni ingrandimento è costituito da un'immagine già memorizzata (niente calcoli) e pronta da stampare. Datosi che anche a memoria non siamo particolarmente dotati, questo stratagemma va in parte modificato. Chi non ha mai giocato da bambino con «le costruzioni», quei mattoncini colorati che potevano facilmente essere incastrati l'uno con l'altro e con un po' di fantasia potevano dar luogo a innumerevoli figure? In un certo senso anche «SPACE HARRIER» del C64 è stato realizzato con

Foto 2 - Una bella Ferrari.

Foto 3 - ... Un gradito ritorno.



Time Crystal.



Zynaps.

«le costruzioni». Abbiamo detto che tutti gli ingrandimenti devono essere già belli e fatti, ma non c'è abbastanza memoria, quindi occorre un set di mattoncini (caratteri) con forme predefinite che opportunamente combinate possono realizzare diverse forme. In memoria, oltre al set di caratteri, ci sono anche le varie combinazioni che danno vita all'ingrandimento. In altre parole gli ingrandimenti sono memorizzati sotto forma di insieme di caratteri. Questa soluzione è estremamente grossolana, perchè un'immagine realizzata con i caratteri non ha molta attinenza al pavimento. Il massimo della fluidità è di 8 pixel. Se l'oggetto è grande il movimento a scatti non dà eccessivamente fastidio, ma quando si tratta di oggetti che sono ancora in lontananza proprio non ci siamo. Anche se il programma sembra realizzato con i piedi, si è tenuto conto di questo inconveniente; si sono rappresentati gli oggetti distanti con sprite. Ricapitolando, un oggetto che appare dall'orizzonte è uno sprite che mano mano cambia forma. Superata la grandezza di uno sprite, l'oggetto diventa un insieme di caratteri fino al momento del massimo ingrandimento. L'inconveniente dell'uso dei caratteri non si limita al movimento scattoso; c'è anche il problema della sovrapposizione dell'immagine che penso conosciate un po' tutti. Un carattere stampato cancella quello che c'era in precedenza al suo posto, quindi se ad esempio due immagini si sovrappongono, quella stampata per prima verrà in parte o totalmente cancellata e fin qui tutto regolare perchè anche nella realtà un oggetto sopra un altro non lascia intravedere quello coperto (tranne se è trasparente). La questione è un'altra. Supponiamo di avere un oggetto di 4×4 caratteri coperto in parte da uno di 2×2 caratteri. Se l'oggetto più piccolo non ricopre pienamente l'area di 2×2 (ad esempio, se è una sfera) la parte vuota andrà a creare dei buchi antiestetici nell'oggetto parzialmente ricoperto (quello 4×4). Nel gioco questo si vede e dà fastidio. Si è cercato di mascherare questi difetti con la velocità, ma non è stata una grande trovata. Tra l'altro la parte grafica ha anche numerose mutilazioni, come il quadrettato (avrebbe richiesto troppa memoria) e tanti altri difetti che non stiamo qui ad elencare.

La cosa che fa più innervosire è che il sonoro è stato eccessivamente trascurato. Senza andare a cercare gli effetti digitalizzati (che «costano») si poteva rimediare con altri effetti che sono pienamente nelle capacità del SID. Si sarebbe certamente aggiunta molto atmosfera, ma quando il gioco si vende lo stesso e non c'è il tempo per le finezze, chi te lo fa fare.

A questo punto volevo farvi notare come, senza eccessivo impegno, potevano ottenersi risultati migliori.

Il listato proposto, nelle pagine precedenti, realizza una sequenza di ingrandimento interamente con sprite. È un po' lunghetto ma, anche se non molto utile, è un bell'esempio.

Per questa prima puntata credo possa bastare, anche se non abbiamo imparato troppe cose.

Concludendo vi rammento che dovete inviare le vostre proposte per il Megagame al solito indirizzo della Redazione.

Buono lavoro e... a presto.

di Marco Pesce

Un videogioco tutto nostro

seconda puntata

Salve a tutti. Come va? Passate bene le vacanze? Le mie, modestamente, sono state ricche di emozioni e di nuove scoperte, ho conosciuto di persona il Presidente degli Stati Uniti e ho fatto un salto in Giappone alla scoperta delle ultime novità informatiche e... bugie! Non è vero niente; trovo estremamente buffo dover parlare di qualcosa che deve ancora accadere, come se fosse già accaduta. In realtà è ancora novembre (siamo agli sgoccioli), ma quando queste pagine saranno imbrattate e lo sprovveduto lettore cercherà di interpretarle... «la befana è già arrivata da un pezzo». Traduzione del delirio: le vostre letterine non sono ancora arrivate.

Lo «spoglio dei pareri» non è stato ancora effettuato. Devo improvvisarmi attappatore di buchi. Cosa siano i «buchi» penso sia chiaro a tutti, ma voglio ugualmente fare uno scemino (ehm... pardon... schemino):

inizio novembre = *stampa su MC di dicembre*
inizio dicembre = *stampa su MC di gennaio*
inizio gennaio = *arrivano le lettere!*

Studiatelo attentamente e se proprio non riuscite a capire che cosa è un buco... fate esaminare la vostra testolina; probabilmente non ci saranno grosse difficoltà per scorgerne qualcuno.

Abbiamo scherzato abbastanza... cerchiamo di fare i seri.

Dunque, chi sono e che cosa voglio avete avuto modo di scoprirlo la volta scorsa. Per tutti quelli che invece non sanno né chi sono (non ce ne importa niente) né cosa voglio (questo già sembra più interessante) vorrei fare un classico «riassunto della puntata precedente». Questa rubrica è dedicata a tutti coloro che credono nella possibilità di realizzare un grande videogioco MADE IN ITALY. Guardandosi in giro è facile rendersi conto che: non siamo capaci o siamo una gran massa di sfaticati. Personalmente sono convinto che sia molto più valida la seconda ipotesi. Certo «sfaticati» sembra ed è un termine un po' azzardato (i motivi sono anche altri). Ricapitolando sembrerebbe un'impresa impossibile, ma... arriva MC. Con questa rubrica ognuno potrà realmente partecipare alla realizzazione di un Super Videogioco.

Tutti i collaboratori entreranno a far parte del team dell'impresa; ognuno vedrà il suo nome all'interno della presentazione del gioco e sulle pagine di MC. È chiaro che i più validi verranno evidenziati, altrimenti nessuno si impegna e viene fuori una schifezza.

La volta scorsa ho invitato i lettori a mandare idee per il gioco, cioè a fare la descrizione di un gioco ideato da loro, badate bene, solo la descrizione a parole, non un listato dell'eventuale programma, quindi non occorre essere programmatori per partecipare. Non esitate a scrivere, magari solo perché credete che «tanto quelli non la leggono neppure la mia lettera». Le vostre lettere verranno tutte, e sottolineo tutte, esaminate! L'invito a scrivere è rinnovato. Mi raccomando, cercate di inventare

qualcosa di nuovo. Basta anche solo un'idea vaga (l'importante è conoscere i vostri gusti).

Nella prossima puntata ci sarà un primo esame dei vari pareri. Nel frattempo anche questo mese ho realizzato qualche modellino che presento nelle foto di queste pagine, questa volta usando la grafica dell'Amiga e dei supporti predefiniti. Oltre alla realizzazione del videogioco questa rubrica si propone di migliorare le conoscenze del programmatore di videogame, più o meno navigato, con una sorta di corso, che comprenderà vari argomenti, tutti collegati con questa «Scienza».

Scuola di Videogame

La volta scorsa siamo stati un po' troppo avari con gli argomenti; ci siamo limitati ad osservare e criticare un videogioco in modo sintetico, senza approfondire troppo. Non è stata una mancanza; è stata solo fatta la scelta di non appesantire troppo la presentazione della nuova rubrica. Tutto ciò per dire che in questo appuntamento ci rifaremo ed entreremo anche nei dettagli degli argomenti proposti.

Parliamo di grafica e nella fattispecie di scrolling. Anche chi non si intende molto di programmazione conoscerà senz'altro questo vocabolo che è molto ricorrente nel linguaggio dei recensori di videogame. Spesso e volentieri si accusano i programmatori proprio per avere trascurato la qualità del suddetto. Per coloro che non ne hanno mai sentito parlare diremo che si tratta di uno stratagemma per ampliare l'area di gioco senza ridurre le dimensioni dei personaggi. Mi spiego. Supponiamo di assistere a una partita di calcio in TV. Il cameraman potrebbe riprendere: l'intero campo, il che comporterebbe delle difficoltà per il riconoscimento dei giocatori, oppure, come di consueto, seguire l'azione di gioco inquadrando solo una parte del campo, tuttavia sufficientemente ampia. Lo scorrazzare dell'inquadratura per il campo, tradotto nel linguaggio dei programmatori viene per l'appunto detto scrolling. I problemi collegati a questa tecnica sono molteplici. Innanzi tutto, facendo un discorso in generale (ovvero rivolto a tutti i computer), quello più noto riguarda la fluidità che solitamente è il fattore che più influenza la qualità. Molti computer non sono stati concepiti per realizzare scrolling e secondo me questa è proprio una grave mancanza. Tuttavia cercano di arrangiarsi e la conseguenza inevitabile è proprio quella di «scarsa fluidità». Prima di proseguire vediamo come due computer differenti, uno dotato e uno non dotato, realizzano questa funzione. Prendiamoli «a caso»: Commodore 64 e Spectrum. Inutile ricordarvi che quello dotato è il C-64 (senza offese per gli Spectri). Quest'ultimo ha nel suo Video-Chip una parte specializzata, ma non troppo. Infatti il VIC-II (non VIC-20... mi raccomando) ha la possibilità di spostare l'immagine visualizzata di un pixel alla volta sia in verticale che in orizzontale, ma solo per 8 pixel. È da notare che questo spostamento è effettuato via hardware e praticamente non richiede un minimo dispendio di tempo. Apparentemente sembrerebbe una cosa da niente, ma diamo un'occhiata allo Spectrum.

Il 48K non ha la possibilità di ricevere aiuti, cioè il povero Z-80 deve caricarsi di tutto il lavoro (... lo Z-80 è il microprocessore). Questo vuol dire che, per spostare di un pixel un'immagine, occorre realmente cambiare la sua posizione in memoria e non avete idea di quanto tempo occorra per realizzare una cosa del genere.

Del C-64 avevamo detto però che l'immagine si poteva spostare di soli 8 pixel con il VIC-II (da qui il «ma non troppo» di sopra). In realtà, tutto ciò è

più che sufficiente, in quanto con questo appoggio è possibile realizzare scrolling fine anche in modo testo (e cioè anche non usando l'hi-res). Il 6510 (... il microprocessore) deve quindi solo spostare un ammasso di 1000 caratteri (1000 locazioni di memoria), mentre lo Z-80 deve trasferire un numero di locazioni ben maggiore (ricordiamoci che siamo in alta risoluzione). Tutto il discorso fatto fin ora può apparire molto oscuro se non facciamo una piccola precisazione. Lo scrolling fluido (o fine...) si ottiene muovendo l'immagine di pochi pixel alla volta, quindi, mentre sul Sessantaquattro, male che vada, occorre spostare 1K di memoria, nello Spectrum, bene che vada, occorre spostare 6K. Lo spostamento, ad esempio, di un pixel alla volta, sul 48K comporta un trasferimento dati troppo imponente e la velocità finisce per rimetterci (effetto rallentatore), quindi per velocizzare si tratta spesso di muoversi a scatti di 8 pixel. Sul Commodore 64 invece, un pixel alla volta non comporta nessun problema, perché dal punto di vista della memoria si tratta di spostare 1K una volta ogni 8 pixel. È proprio per questo che sul Commodore ci sono stupendi effetti mentre il povero Sinclair è tutto un terremoto.

Qualcosa mi fa supporre che tra non molto il confronto potrebbe degenerare con tristi conseguenze; alla mia vita ci tengo quindi lasciamo stare lo Spectrum e occupiamoci a fondo del nostro 64.

Mi sono fatto prendere dall'euforia e in sostanza non ho ancora chiarito come avviene uno scrolling sul C-64. Se è vero che vogliamo spostare un'immagine di un pixel alla volta vediamo passo passo che cosa accade nel computer. Supponiamo che sul video sia visualizzato un gruppo di caratteri ridefiniti che insieme rappresentano, ad esempio, un palazzo. La prima operazione da fare consiste nello stringere lo schermo visibile (allargando il bordo), di un carattere in verticale e di due in orizzontale (poi vedremo perché). Sullo schermo ora sono visibili 38×24 caratteri (fig. 1), ma in realtà ci sono ancora tutti. Praticamente è come se avessimo attaccato quattro strisce di nastro isolante sul monitor; due strisce verticali che coprono la prima e l'ultima colonna e due strisce orizzontali che coprono mezza prima e mezza ultima riga (cioè solo una riga). Quello che abbiamo impostato resterà sempre così per tutto il periodo di scrolling. A questo punto interviene il VIC-II. Modificando due registri (orizzontale e verticale) l'immagine viene di conseguenza spostata. Il «passo», come abbiamo detto, è di un pixel, il numero di passi è 8 (sempre «come abbiamo detto»), quindi bastano solo 3 bit per ogni direzione, che presi da un byte lasciano 5 bit x 2 inutilizzati. In verità sono tutt'altro che inutilizzati, ma per ora lasciamoli stare. A proposito, il registro orizzontale è il numero 22 (i primi tre bit) e cioè corrisponde alla locazione 53248+22, quello verticale è il numero 17 (sempre i primi tre bit) e cioè la locazione 53248+17. Torniamo al nostro palazzo che sta aspettando. Se vogliamo, ad esempio, scrollarlo verso sinistra, il registro verticale non ci interessa e quindi non lo «toccheremo». Occorre invece decrementare quello orizzontale fino ad azzerarlo. Cosa succede ora? Il VIC-II non ce la fa più! Interviene il microprocessore che a questo punto deve spostare tutti i caratteri di un posto verso destra. Dato che un carattere corrisponde a 8 pixel, se rimpostiamo al massimo valore il registro orizzontale e lo decrementiamo di uno, come se niente fosse successo, abbiamo praticamente realizzato uno scrolling di un pixel in aggiunta agli 8 precedenti (ottenuti con il solo VIC-II). Tradotto in altre parole, con questo sistema possiamo realizzare scrolling per tutti i pixel che vogliamo. Il diagramma di flusso di figura 2 riassume quanto detto fin ora.

Resta da chiarire che fine fa il palazzo se arriva al margine destro dello schermo.

Uno scrolling di questo tipo può essere realizzato in due modi:

1) con rotazione
2) continuo.

Se continuassimo a spostare il nostro palazzo, arriverebbe il momento di farlo uscire dallo schermo. Finora infatti non abbiamo tenuto conto di un piccolo par-

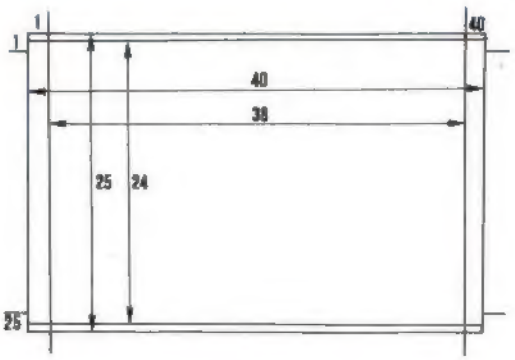


Figura 1
Dimensioni dello schermo prima del restringimento (40×25) e dopo (38×24).

Figura 2

Figura 3

ticolare: i caratteri della prima colonna non possono essere spostati oltre. Ci sono due possibilità: eliminarli, oppure riposizionarli nella quarantesima colonna. Quest'ultima possibilità equivale proprio al primo modo: comporta una visione rotatoria, cioè quello che esce da un lato rientra dal lato opposto. Pensandoci bene però, in questo modo non abbiamo per niente ampliato l'area di gioco. Lo scrolling più importante è quello continuo, ovvero quello che dalla parte opposta a quella di uscita fa entrare altri dati, ad esempio, un altro gruppo di caratteri ridefiniti rappresentanti un albero. Quindi oltre alla routine di scrolling ne occorre una che immetta i nuovi «dati». Il diagramma di flusso di figura 2 in tal caso va modificato come quello di figura 3.

A questo punto facciamo la conoscenza di un altro problema collegato allo scrolling: lo «sfarfallamento». Diremo che deriva da un mancato sincronismo e mi accorgo che a questo punto occorre prima chiarire come il VIC-II visualizza un'immagine.

Come forse già sapete, il quadro del monitor viene visualizzato da un pennello elettronico ovvero da un «pixel» che parte dall'angolo superiore sinistro e termina la sua corsa nell'angolo inferiore destro. Il pennello avanza orizzontalmente e arrivato al margine destro torna di colpo a quello sinistro ma una righina più sotto (fig. 4). Procedendo con questo metodo giunge fino all'ultimo pixel per poi ricominciare immediatamente da capo. Il pennello ha una velocità elevatissima, tant'è vero che l'occhio umano non si accorge di nulla e vede solo l'immagine nel suo insieme. Per quanto possa essere veloce però, un certo tempo lo impiega, ma come ci accorgeremo più avanti, non è affatto una limitazione (anzi!).

Enunceremo ora un piccolo difettuccio del Video-Clip. Il caro VIC-II è un po' schizzinoso in quanto fa il seguente ragionamento: se «qualcosa» non è stata visualizzata dall'inizio perché visualizzarla in parte? Bella volpata! Vediamo cosa comporta questa discriminazione. Prendiamo ad esempio uno sprite che si sta muovendo verso l'alto; se il pennello elettronico ha già percorso la riga superiore a quella dov'è posizionato in questo momento lo spirte, ma non è ancora arrivato a visualizzarlo e disgraziatamente proprio in quel momento lo sprite viene spostato, indovinate cosa succede? Quando il pennello dovrebbe visualizzare lo sprite si accorge che la prima riga non è stata visualizzata e quindi ignora la sua presenza. Otteniamo così il fantastico effetto sfarfallamento che è la croce dei programmatori Basic sul C-64.

Tornando a noi, qualcosa di simile accade anche se ci accingiamo ad effettuare uno scrolling verticale ed è molto più fastidioso che nello sprite (tutto lo schermo che sfarfalla).

Per rimediare a tutti gli sfarfallamenti c'è un ottimo stratagemma. Fortunatamente tra i registri del VIC-II ce n'è uno che permette di testare la posizione del pennello elettronico. Come è facile intuire basta effettuare qualunque aggiornamento grafico quando il pennello visualizza, ad esempio, il bordo, per risolvere il quesito.

Riprendiamo il discorso sulla fluidità. Anche se potrebbe sembrare una contraddizione a quello che abbiamo detto finora, non occorre spostare l'immagine di un pixel per volta per ottenere scorrimenti fluidi. Seguite questo ragionamento. Se sposto l'immagine di un pixel alla volta occorre anche che questo avvenga ogni qualvolta il Video-Chip termina di visualizzare un quadro. Questo significa che teoricamente la frequenza di aggiornamento del quadro equivale al numero di pixel «scrollati» al secondo. Tutto ciò perché siamo entrati nell'ottica che «se non si muove di un pixel alla volta non è fluido». Volete sapere l'ultima? Anche a scatti di 8 pixel può essere fluido! L'importante è essere sincronizzati con l'apparizione dell'immagine video. Tutto ciò equivale a dire che un'immagine deve essere aggiornata in modo ritmico (cioè, tante sono le visualizzazioni del VIC-II, tanti sono gli aggiornamenti), altrimenti anche con uno spostamento di un pixel alla volta si ottiene la famigerata «scarsa fluidità». Infatti fluido non vuol dire solo continuo, ma anche e soprattutto a velocità costante. In definitiva, aggiornando l'immagine in sincronismo con il pennello elettronico otteniamo:

1) assenza di sfarfallamenti
2) velocità costante (= fluidità).

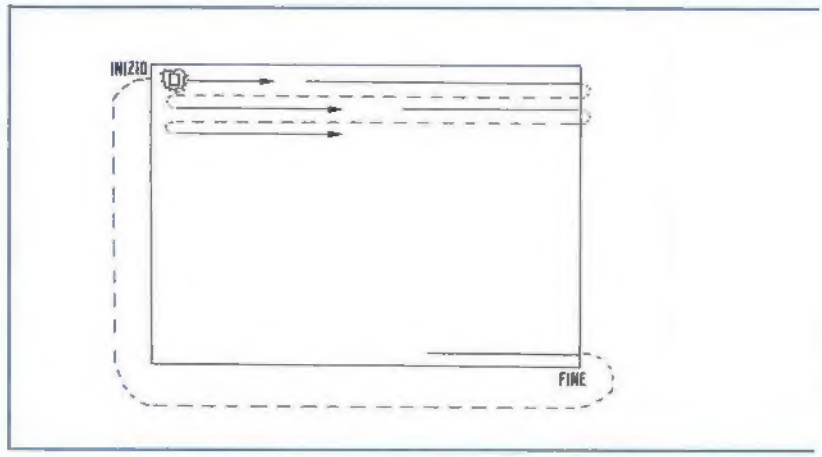
Quindi anche muovendosi di 8 pixel in 8 pixel lo scrolling sarà fluido. A questo punto immagino le vostre facce incredule. Certamente vi chiederete: «ma allora a cosa serve un movimento pixel per pixel?».

Risponderò semplicemente: «per andare più lentamente». Infatti se è vero che di 8 in 8 pixel lo scrolling in queste condizioni è fluido, è anche vero che sembra di andare su una Ferrari. Ricapitolando, spostando l'immagine di un pixel alla volta in sincronismo con il pennello elettronico, otteniamo uno scrolling a velocità moderata e fluido, mentre spostando l'immagine di più pixel alla volta, sempre in sincronismo con il pennello, otteniamo uno scrolling a velocità proporzionale al numero dei pixel ma sempre fluido!

Chiuso il discorso sulla fluidità.

Sullo scrolling invece c'è ancora molto da dire. Quello che abbiamo finora descritto è un tipo di scrolling, ma ne esistono anche degli altri. Inoltre non abbiamo ancora parlato di programmazione. Come avrete intuito, tutto ciò sarà trattato nella prossima puntata. Per ora non mi resta che salutarvi e ricordarvi che se siete interessati ad un particolare argomento o avete qualche critica da fare, basta scrivere una lettera; cercheremo di accontentare tutti. Ciao!

Figura 4
Quando il pennello elettronico arriva al margine destro, torna al «capoverso» senza «scrivere» niente.



Chiunque desidera partecipare a questa iniziativa può inviare il suo contributo alla redazione di MCmicrocomputer. Saranno molto gradite anche osservazioni e critiche. Il «materiale» da inviare sarà di volta in volta descritto nel testo della rubrica.

di Marco Pesce

Un videogioco tutto nostro

terza puntata

Che cosa vi aspettate da questa puntata? Scommettete che indovino? Volete sapere cos'è successo con l'arrivo delle lettere... ■ non dite che non ho indovinato (... per favore)

Bene, sono arrivati circa 15-16 TIR pieni zeppi di lettere diretti ■ questa rubrica.

In poco più di otto minuti ■ ventisei secondi abbiamo letto attentamente tutt'e 57282749 lettere e qui di seguito abbiamo riportato le proposte più interessanti. Esattamente come dire. . che le lettere arrivate dall'uscita di MC di dicembre a quando io ho dovuto consegnare questo articolo non erano poi moltissime. Tutte le altre lettere (quelle nei TIR che prima o poi arriveranno, vero?) le analizzeremo per il prossimo numero. Vediamo ora la proposta di un lettore di Genova, Paolo Costabel:

«... Una corsa ad ostacoli, con effetti tridimensionali, scrolling differenziato, giganteschi sprite di razza, scalpitare di zoccoli digitalizzato, centinaia di percorsi diversi, possibilità di giocare in più persone conteporaneamente, opzionalmente via modem ecc. ecc...».

Già si tratta proprio di una simulazione ippica.

Più dettagliata ■ la proposta di altri due lettori, Fabrizio Baldini e Stefano Innocenti che ci scrivono da Vinci:

«... Il genere ■ poliziesco. Il protagonista, invece di essere un paladino della giustizia, è uno sporco KILLER che non esita ad uccidere quando riceve un bel gruzzolo. Il giocatore quindi dovrà stare attento a non farsi acciuffare dalla polizia e naturalmente ■ non fallire la sua missione. L'azione si svolge in una metropoli statunitense. All'aumentare dei guadagni ■ può ampliare l'armamento bellico del personaggio. Le sequenze di movimento del giocatore potrebbero essere realizzate sul tipo di quelle di JACK THE NIPPER. Naturalmente il programma non sarà del tutto un arcade, ma includerà alcune sequenze adventure nelle quali si sceglierà se accettare nuovi incarichi, comprare armi, ecc. ecc. Il Killer ha a sua disposizione una opzione che gli permette di avere le informazioni necessarie sulla vittima (cioè dove abita, a quale ora torna ■ casa la sera ecc. ecc.), poi sarà lui a decidere quando e come ucciderla, anche se avrà a disposizione un tempo limitato. In ogni caso la polizia potrà intralciare i piani del giocatore perquisendo la sua abitazione, indagando su di lui ■ scortando la probabile vittima. Il gioco termina quando il

giocatore fa una brutta fine oppure quando diventa talmente ricco e potente da essere lui il BOSS...».
Sentiamo cosa hanno da dirci Flavio, Fulvio ■ Stefano Chini di Poggio Mirteto Scalo:

«... un simulatore di motociclismo. Il gioco consisterà in una gara più realistica possibile. Sullo schermo dovrà figurare: nella parte bassa il contagiri non digitale, la pressione dell'olio e la marcia. Nella parte alta dello schermo dovrà apparire il cronometro. Lo scenario consisterà in una strada con i cordoli e senza linea centrale...

Dulcis in fundo un certo Gianni Z.:

«... TITOLO: Viaggio nel tempo.
NUMERO GIOCATORI: 1
GRAFICA: 3D
TRAMA: uno dei più grandi scienziati dei nostri tempi, ovvero me stesso, inventa e sperimenta una macchina del tempo.

Essa però (mannaggia a me!) non funziona a dovere ■ mi proietta nel passato in qualche epoca misteriosa... Chi mi salverà? Un cavaliere del tempo preso dalla redazione di MC!! Il quale dopo aver scoperto il guasto della macchina del tempo, parte alla mia ricerca (a questo punto ci conto poco). Tale cavaliere dovrà combattere contro vari nemici sino a quando non mi troverà e mi salverà!
SVOLGIMENTO:
Schermata iniziale: presentazione cavaliere del tempo e relativa macchina del tempo.

Il gioco: si svolge in variate epoche ■ con le armi del tempo considerato. Il protagonista non termina una battaglia nel modo classico, ovvero ammazza tutti, fa un macello e poi cambia schermo, ma, al contrario, il viaggio nel tempo fa sì che il cambiamento avvenga all'improvviso ■ senza spargimenti di sangue. Esempio: il gioco inizia nell'antica Roma con un combattimento di lotta Greco Romana, poi all'improvviso il contesto cambia e ci troviamo nel Vietnam. Successivamente ci potremmo trasferire in Cina ecc. ecc. Il cambiamento d'epoca avviene in maniera veloce, ma non troppo brusca; l'impressione deve essere che il paesaggio subisca una specie di metamorfosi...».

Fine delle proposte.

Analizziamo la situazione. La prima proposta, cioè quella ippica (guardate il modellino della foto in prima pagina), riguarda un tema poco sfruttato e, data anche la sua particolare impostazione, sufficientemente interessante. I lettori di Vinci la pensano in maniera differente e si riversano su un tema ben collaudato (forse anche troppo) Flavio, Fulvio e Stefano, con la loro concisione, non hanno aggiunto niente di nuovo, o quasi, al modello SUPERCYCLE.

Ho l'impressione che non ci siamo molto sforzati con la fantasia. Non voglio con questo farvi un rimprovero, ma son sicuro che un gioco per essere bello deve anche avere una concezione originale. Da questo punto di vista il lettore Gianni Z. (ma ti costava così tanto scrivere anche il cognome intero?) ha colto nel segno e si merita il modellino dell'altra foto in prima pagina. In ogni modo io avevo detto di mandare solo delle idee, per conoscere i gusti dei lettori, quindi...

Scuola di videogame
Scrolling 3D

Puff... puff... non ce la faccio più... sto correndo come un matto per finire questa puntata prima di lunedì... speriamo di farcela... Velocemente il riassunto della scorsa volta.

Dunque parlavamo di scrolling, quello classico ovvero senza effetti parallattici tridimensionali che invece saranno l'argomento di questa puntata, abbiamo anche brevemente descritto come avviene la visualizzazione di uno schermo video e perché si verificano gli sfarfallamenti e ancora si è parlato della fluidità.

Continuiamo quindi il discorso analizzando una nuova categoria, quella degli effetti 3D.

Lo scrolling visto fin ora ha un piccolo particolare che lo differenzia dalla realtà. Se infatti durante un viaggio in macchina osserviamo il panorama dal finestrino laterale, quello che vediamo è per l'appunto uno scrolling, ma a dispetto di quello che succede con uno scrolling computereccio, gli «oggetti» in lontananza si muovono più lentamente e quelli molto lontani addirittura sembrano fermi. Come fare per realizzare questo con il computer? Semplice, basterebbe creare dei piani, ognuno con oggetti a distanza diversa ■ farli scrollare con velocità differente. Seguite la figura 1a e tutto sarà più chiaro. Sovrapponendo poi i piani (nel nostro caso sono tre) otteniamo il risultato di figura 1b. A questo punto se il piano 2 ha uno scrolling praticamente nullo, il piano 3 ha uno scrolling ■ media velocità e infine il piano 1 uno scrolling ad alta velocità, ■ ottiene l'effetto 3D della vista dal finestrino della macchina. Tutto qui. Ma... un momento... come realizzare la sovrapposizione dei tre piani? Risposta: non è un'impresa facile. Questo perché il 64 non può effettuare un'operazione del genere in modo autonomo. Occorre infatti simulare la sovrapposizione oppure utilizzare gli sprite. Vediamo cosa voglio dire.

Simulare la sovrapposizione, lo dice il nome stesso, vuol dire che in realtà non ci sarà sovrapposizione. Ci sarà invece uno scrolling a più piani suddividendo lo schermo in settori. Ancora una volta una figura (fig. 2) è più chiara di molte parole. Il settore 1 rappresenta il piano 2, il settore 2 rappresenta il piano 3 e infine il piano 1 è rappresentato dal terzo settore. Apparentemente non c'è stato un gran cambiamento. Anche un bambino di due anni (... quasi) capisce a questo punto che per ottenere l'effetto dei tre piani basta far scrollare a diverse velocità i tre settori:

bassa velocità = settore 1
media velocità = settore 2
alta velocità = settore 3

Anche dal punto di vista grafico le due soluzioni si somigliano (in questo caso molto). Tuttavia se i tre piani avevano parti che si sovrapponevano eccessivamente, era più opportuno adottare la soluzione «usa sprite». Questo secondo metodo è da utilizzare con

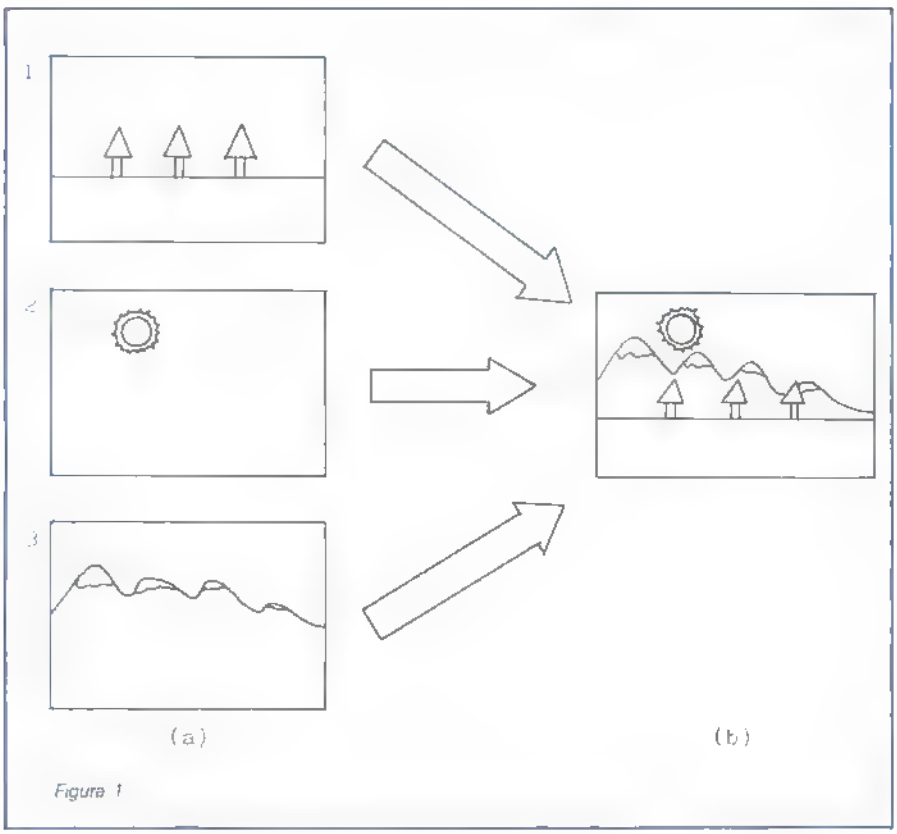


Figura 1

cautela in quanto impegna parecchi sprite. In ogni caso, con la suddivisione a raster si risolve anche questo problema (dopo spieghiamo cos'è la «suddivisione a raster»). Vediamo in cosa consiste l'utilizzazione degli sprite. Se ad esempio ci trovavamo nella condizione di figura 3 la sovrapposizione dei piani avrebbe comportato l'effetto «si vede-non si vede» (fig. 4) che con i settori non si può imitare. Con gli sprite invece sì. Basta realizzare qualche piano per l'appunto usando sprite. Infatti il piano 1 di figura 3 non è eccessivamente complesso e può essere sostituito con uno sprite (o più di uno) avente la sua forma. Anche il piano 3, volendo, può essere sostituito con sprite a questo punto il gioco è fatto. Dicevo prima che occorreva essere cauti con questa tecnica ed è facile dimostrare che ho ragione perché, se la scena si fa complessa (anche il nostro caso non è male), la situazione «sprite ancora liberi» diventa critica. Ci viene in aiuto il sistema di suddivisione a raster accennato in precedenza. Esso consiste nell'utilizzare più volte lo stesso sprite. Ciò è possibile ed è assolutamente normale, se la riutilizzazione avviene dopo la fine della prima visualizzazione (!?!!?). Prima di farvi impazzire vi suggerisco di osservare la figura 5. Ve-

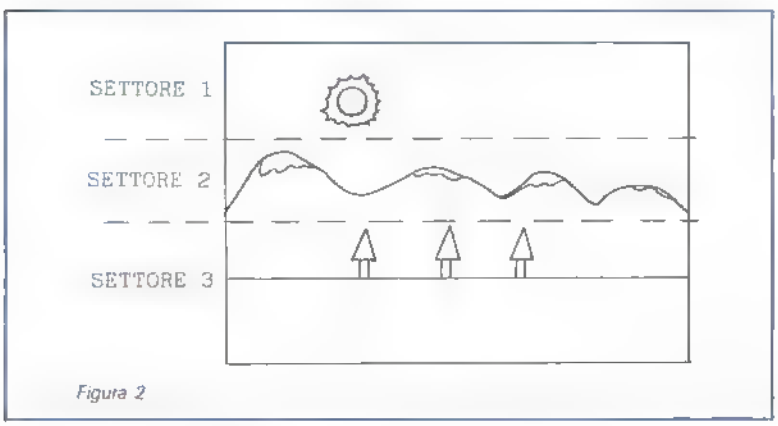


Figura 2

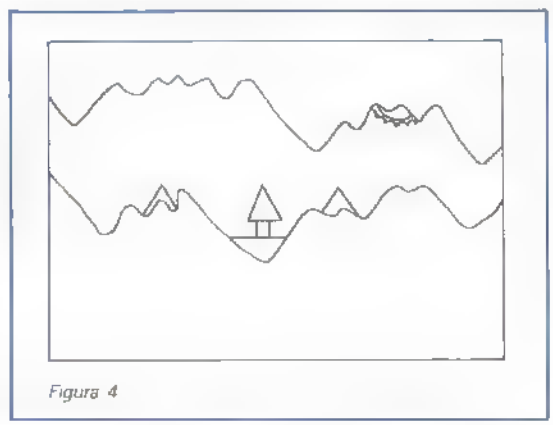


Figura 4

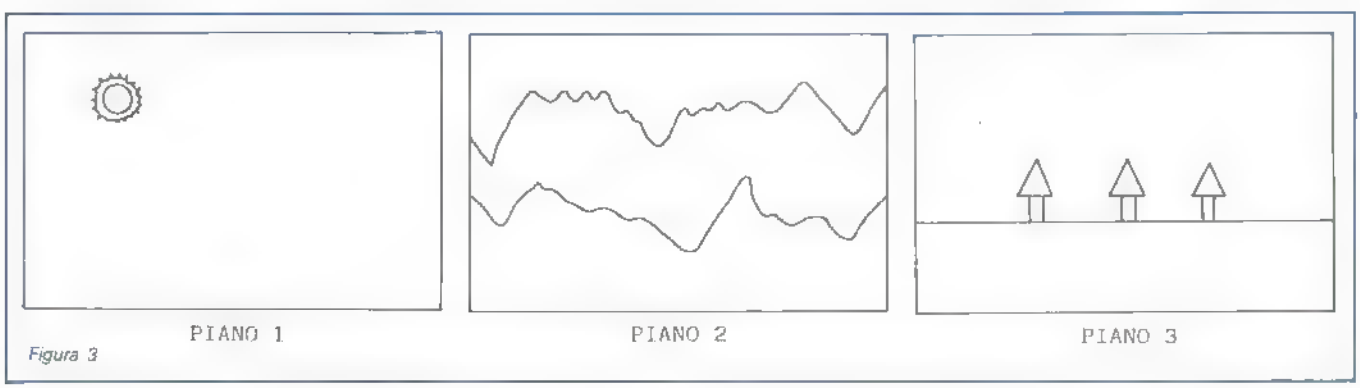


Figura 3

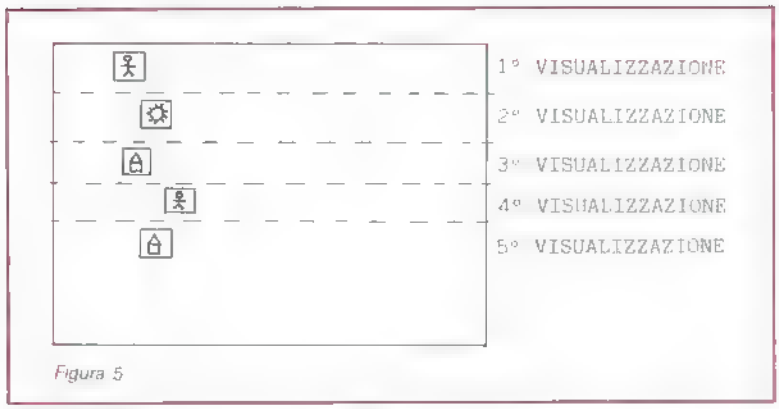


Figura 5

diamo, in ordine, partendo dall'alto, un omino e un sole, una casetta, ancora un omino e ancora una casetta. Questi cinque oggetti sono visualizzati facendo uso di un solo sprite. Come si fa? Semplicissimo, si impostano le coordinate del primo oggetto da rappresentare (l'omino) e la sua forma, poi arrivati a fine visualizzazione, ovvero quando l'omino è stato disegnato completamente dal pennello elettronico (cfr. la scorsa puntata), lo sprite viene spostato e vengono impostate le coordinate e la forma del secondo oggetto da visualizzare (il sole). Terminata anche la sua visualizzazione, possiamo passare al terzo oggetto e così via. Come facciamo a sapere quando è stata completata la visualizzazione di un oggetto-sprite? La volta scorsa avevo parlato del pennello elettronico e del registro (+ un bit, cioè 9 bit) che ne contiene la posizione (d'accordo lo avevo solo accennato); basta testare la posizione del pennello e spostare lo sprite al momento giusto. L'ipotetica routine che realizza quanto detto

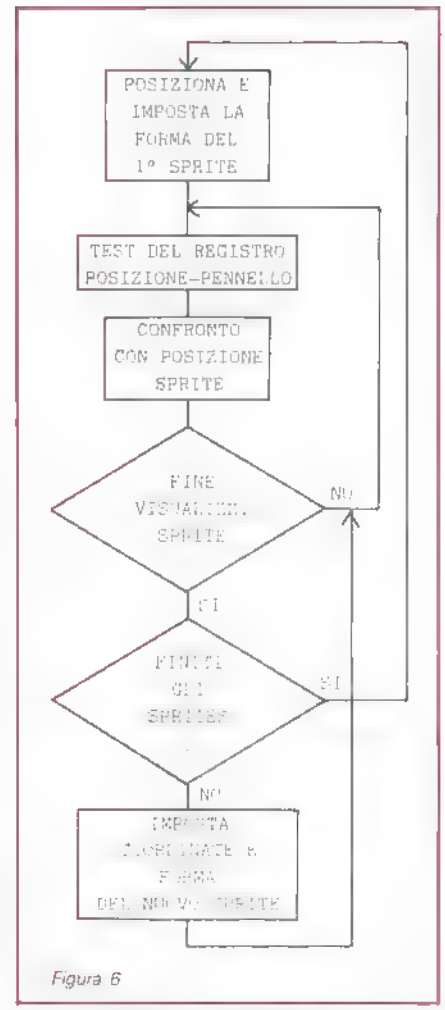


Figura 6

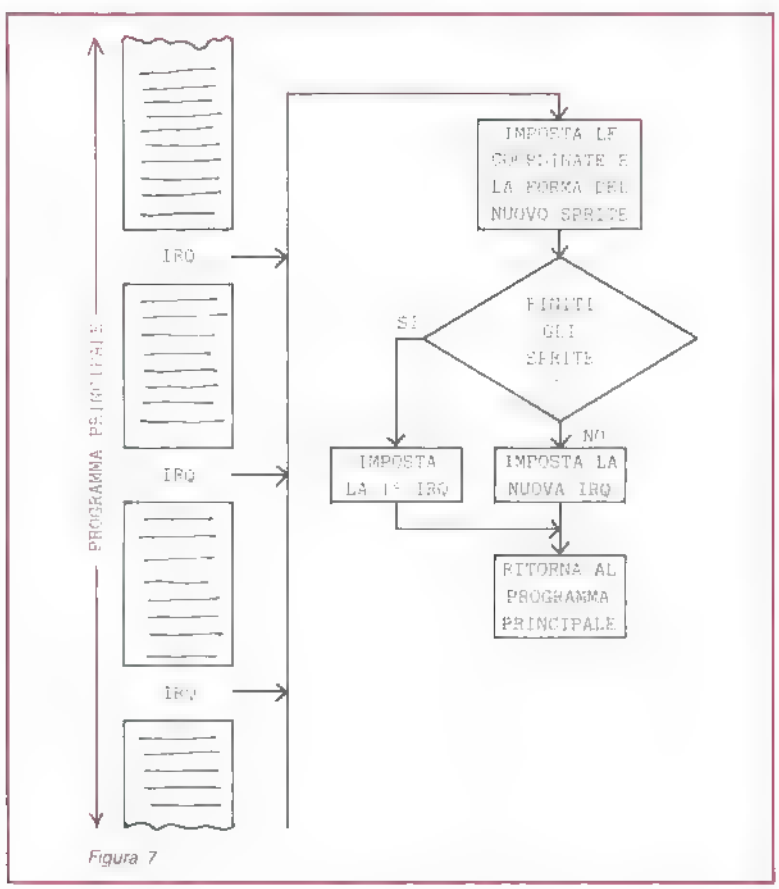


Figura 7

sopra deve essere del tipo rappresentato in figura 6 (dove vediamo il diagramma di flusso). Essa posiziona il primo oggetto-sprite e controlla la posizione del pennello elettronico, confrontando il registro che lo rappresenta con il punto in cui lo sprite finisce di essere visualizzato (21 righe-video più sotto della coordinata y).

Quando il confronto è un'uguaglianza si controlla che gli sprite-oggetto da visualizzare non siano finiti e in caso affermativo si imposteranno le coordinate e la forma del nuovo sprite-oggetto. Se invece gli sprite-oggetto sono finiti si ricomincia da capo con il primo sprite.

Una routine del genere però impegna in modo totale il microprocessore in quanto il ciclo di test della posizione del pennello elettronico non può essere abbandonato e deve essere eseguito il più velocemente possibile. È chiaro quindi che se, contemporaneamente alla visualizzazione dello sprite moltiplicata, vogliamo effettuare qualche altra opera-

zione, bisogna trovare una soluzione differente. Fortunatamente ci vengono in aiuto le famose IRQ. Se non sapete cosa sono non preoccupatevi; torneremo presto sull'argomento. Per ora diciamo che tramite il loro utilizzo è possibile controllare la posizione del pennello elettronico senza impegnare il microprocessore in dispendiosi «cicli». Basta impostare la posizione del pennello che ci interessa e al resto penserà l'IRQ. Al momento opportuno infatti, qualunque operazione sia in corso, il microprocessore la abbandonerà e eseguirà una particolare routine. Quest'ultima routine può essere programmata, quindi, seguendo questa filosofia la routine di fig. 6 può essere trasformata in quella di figura 7.

Vediamo in dettaglio che cosa succede.

Inizialmente occorre impostare la prima IRQ in corrispondenza della coordinata y del primo oggetto-sprite da visualizzare. Inoltre viene impostato l'indirizzo della «routine IRQ» da noi programmata (quella rappresentata come diagramma di flusso in figura 7). Durante queste operazioni le IRQ vengono disabilitate, in modo da non far avvenire una IRQ involontaria proprio mentre le stiamo eseguendo (un disastro). A questo punto possiamo eseguire il nostro programma principale senza minimamente preoccuparci degli sprite-oggetto. Al verificarsi delle IRQ il programma principale verrà momentaneamente sospeso. Sarà eseguita allora la nostra benedetta routine IRQ, che posizionerà il nuovo sprite-oggetto, imposterà la nuova posizione del pennello per la prossima IRQ e terminerà con il ritorno al punto esatto in cui era stato sospeso il programma principale. Alla prossima IRQ ■ posizionerà il nuovo sprite-oggetto ■ così via.

Certo che se istituissero un premio per il più divagante lo vincerei senza difficoltà. A proposito di raster, per effettuare la suddivisione in settori della figura 2, occorre utilizzare proprio la tencica delle IRQ. Suppongo che a questo punto siate stanchi (bella scusa questa... quasi quasi me la rivendo) e non vi va di sapere come realizzare tale tecnica. A parte gli scherzi, il tempo stringe e vi lascio... anzi leggetevi prima la megaposta.

Megaposta

A cosa serve la Megaposta? Risposta: serve a discutere sulle proposte fatte dai lettori per quello che riguarda la rubrica in generale e ■ risolvere i problemi che possono sorgere.

Le stesse lettere che parlano del Megagame ci suggeriscono delle proposte che ora andremo ad esaminare.

Il lettore Paolo Costabel di Genova ci dice:

«... Se plaudo allo spirito dell'iniziativa, sono piuttosto scettico sulla possibilità di lavorare per lettera. Non sarebbe meglio passare attraverso MC-Link?.

... pensavo di realizzare un programmino di conversione da IFF a 64 per poter utilizzare Dpaint per disegnare sprite, sfondi, titoli etc. per il 64.

L'unico problema per me sono i suoni, dei quali conosco poco. In ogni caso anche per gli effetti sonori vale il discorso di prima; si potrebbe convertire un file IFF con le ovvie limitazioni, e usare programmi di Amiga del livello di DMCS o digitalizzatori tipo Future-Sound.

In ogni caso se la cosa funzionerà, potrebbe essere il punto di partenza per una specie di software house MC-lettori. Scommetto che ti arriveranno un sacco di buone idee. Perché fermarsi a un solo videogioco? Vedrei molto bene un adventure alla Infocom o software serio (avevo cominciato a lavorare a un compilatore Fortran, linguaggio di cui non esiste una versione per C-64)...»

Non hai tutti i torti dicendo che sarebbe meglio «passare attraverso MC-Link», tuttavia non credo che la maggior parte dei lettori sia in possesso di un modem; ne riparleremo in seguito.

L'idea di utilizzare il Dpaint per la grafica del sessantaquattro non è male e a dire il vero era venuta anche a me.

In effetti si vede che non ti intendi molto di suoni sul Commodore; il 64 non è in grado di realizzare musica digitalizzata sul modello di quella di Amiga. È vero che si sentono a volte suoni digitalizzati, ma sono fatti impegnando il microprocessore e non permettono di far girare contemporaneamente anche il resto del programma. Anche di questo torneremo a discutere. Chissà che l'idea di «più Megagame al prezzo di uno» non venga anch'essa presa in considerazione...

Sentiamo Stefano Flavio ■ Fulvio:

«... Noi desideriamo che lei (tramite MC o per posta) ci invii un semplice esempio di una strada in movimento per C128 o C64...»

D'accordo, probabilmente avrete il vostro esempio, ma per ora non posso accontentarvi.

Infine vediamo cosa ci propone Gianni Z.:

«... Vorrei ora proporvi un giochino che a me ha fatto perdere la testa:

Due signori si incontrano dopo alcuni anni e mentre parlano del + e del −, uno dei due chiede all'amico l'età dei suoi figli. Questo gli risponde con la classica frase — Prova ad indovinare... Il signore gli chiede un suggerimento e l'amico gli risponde che il prodotto delle tre età dà come risultato 36.

Di nuovo l'amico chiede un altro suggerimento poiché il primo non gli è sufficiente. Gli viene detto che la somma delle tre età dà come risultato il numero di casa che questo vede di fronte a sé!

L'amico insiste dicendo che gli indizi non sono sufficienti.

Come ultimo suggerimento gli viene detto che l'ultimo dei figli ha gli occhi azzurri!

A questo punto il prode signore ha tutti gli elementi necessari per fornire le tre età.

Quali sono le tre età? E come fa a scoprirle?...»

Sinceramente non mi importa molto (in realtà è solo una scusa), ma penso che i lettori siano curiosi di sapere la soluzione... Ti prego inviacela!!!

Con quest'ultima Pesciata vi saluto. Ciao ciao! MC

di Marco Pesce

Un videogioco tutto nostro

Quarta parte

Finalmente ci siamo. Sono arrivate le tanto attese lettere e adesso ve le sorbite tutte. Detto in altre parole, in questo numero dobbiamo esaminare il parere della gente e quindi ci leggeremo insieme tutte le proposte.
Cominciamo subito? ... Sì.

14 gennaio.
Arriva la lettera di Daniele Argento

«Hai letto — The Hobbit — di J.R.R. Tolkien?

Narra della prima avventura di un giovane Hobbit, Bilbo Baggins, alla caccia di un tesoro situato nella montagna dei N..., espugnata anni prima dal drago Smang.

Che ne diresti di trasformarla in game?»

Sbaglio o questa storia si è già sentita altre volte?

15 gennaio. Umberto Michelucci

«Voi muovete un elicottero visto dall'alto (tipo — Tuono blu — per intenderci) e vi spostate dentro una città (!), si avete capito bene, dentro una città (so già che sarà difficile da realizzare...). Portando il joystick verso destra vi ruotate in senso orario e portando il joystick verso sinistra vi ruotate in senso antiorario. Il bottone servirà per sparare con la mitragliatrice e il joystick messo in avanti e indietro rispettivamente per accelerare o decelerare. Questo è tutto quello che mi viene in mente (scusate se è poco)».

D'accordo sei scusato.

21 gennaio.
Due proposte da Franco Violante

«Siamo in una galassia formata da una serie di sistemi solari. Su alcuni pianeti gli uomini hanno sviluppato le loro civiltà, poi si sono sparsi sugli altri pianeti dello stesso sistema, creando basi militari e commerciali o trasformando i suoli e le atmosfere per la vita umana.

All'inizio del gioco abbiamo così una mappa della galassia con i vari sistemi solari, pochi abitati e molti no. Scopo del giocatore è di creare un impero partendo da un solo sistema a sua scelta, combattendo numerose battaglie (e la lista dei tipi di armi è lunga) nello spazio e sui pianeti, creando alleanze, subendo tradimenti o facendone, stabilendo utili rapporti commerciali con altri pianeti, sfamando le popolazioni dei loro domini, producendo cibo ed estraendo i metalli fondamentali ai suoi scopi militari o commerciali.

Oltre questo c'è la possibilità di essere al comando di una delle varie navi da battaglia (incrociatori leggeri, corazzate, caccia etc.) durante un combattimento. Così entrano in gioco i simulatori di volo e di combattimento, da ricche strumentazioni e possibilità nel caso di semplici caccia spaziali a grandi responsabilità e capacità nel caso di gigantesche stazioni spaziali militari o di grandi corazzate.

Per chiarire l'aspetto generale: è un wargame, con la possibilità di partecipare ai combattimenti con i simulatori, con una grande scelta di armi, con situazioni di tipo economico, commerciale e politico.

Nel caso questa mia idea venisse accettata, ti dico che essa è già in parte realizzata (al 50%), ma non vi posso dare alcun aiuto pratico in quanto non so assolutamente niente di programmazione — et similia —. Passiamo ora alla seconda idea.

Essa è completamente diversa dalla prima, sebbene anche in questa ci sia un bel po' di fantasia.

Siamo in una terra immaginaria e fantastica, abitata da numerosi popoli molto diversi tra loro, densa di segreti, ma anche colma di ricordi del passato.

In una parte di questa terra vive un popolo, i Wantorrior, giunto nella loro terra da vari secoli, ma originario di una terra molto lontana, su a nord. Questo popolo, in tempi antichi, emigrò da quella terra, si spinse verso sud e giunse in quella attuale passando attraverso una infinità di terre diverse, abitate e non, ostili o pacifiche, meravigliose o orrende.

Dopo una lunga marcia trovarono un bel territorio, confortevole e disabitato, in cui subito si installarono.

Passarono molti secoli e intanto la loro terra veniva inghiottita a poco a poco dal deserto meridionale. Venne così il giorno in cui per i Wantorrior era il momento di andare via.

Dopo questa breve storia, ecco il gioco: il popolo che emigra verso le terre che abitava anticamente. I Wantorrior sono un popolo particolare, formato da 10 tribù, ognuna con una propria mansione all'interno della comunità. Essi attraverseranno territori ricchi di nemici, di sorprese, di scoperte, di aiuti insperati, di dolori e di sofferenze, di guerre e di paci etc. Scopo del giocatore è portare i Wantorrior nella loro terra, portarvene quanti più ne può (molti moriranno a causa delle difficoltà del viaggio), portarveli quando prima possibile.

All'inizio il giocatore potrà conoscere approfonditamente le caratteristiche delle 10 tribù; avrà una conoscenza dettagliata solo dei territori confinanti il suo, una mappa generale (scritta dai Wantorrior nel loro primo viaggio) risalente appunto ad un'epoca lontana, quindi intanto le cose saranno cambiate. Nel gioco la magia è molto presente (incantesimi, oggetti magici etc.) così pure l'imprevisto e lo sconosciuto. Tutto tenuto insieme da una sua storia.

Questa idea è realizzata al 35% e vi posso dare qualche aiuto (sempre teorico) sui comandi e possibilità del giocatore.

Sarebbe molto utile, per arricchire di più il gioco, realizzare delle scene animate, nel maggior numero possibile.

Ho detto prima che queste sono le due idee che ho meglio elaborato; ce ne sono altre, infatti, ancora allo stadio embrionale, mentre devo dire che una terza idea completamente realizzata è già in mano ad un mio amico che sta cercando di realizzarla sul computer, procedendo molto lentamente, a causa della sua mancanza di tempo. Anche questa idea appartiene al genere fantasy, ma è molto diversa dall'altra qui esposta. Nel caso tu lo richieda posso esporti anche quest'altra»

Questi probabilmente potrebbero essere due discreti MEGAGAME, ma sinceramente mi sembrano un po' troppo particolari.

22 gennaio. Italian Team (un gruppo di cinque ragazzi)

«Il videogioco parla di un diciottenne che ha per moto un cinquantino tutto scassato e con questo deve vincere una gara contro due Aprilia F1 nuove fiammanti, su di un percorso asfaltato e quindi in questo modo sembra che per Fafo (questo è il nome del diciottenne) le speranze siano poche, ma Fafo è molto sfortunato ed emana molta sfortuna, tanta che le Aprilia alla fine avranno qualche guasto e Fafo potrebbe vincere nonostante incontri molte cose insolite (tipo carriarmati, greggi di pecore, aerei che gli lanciano bombe), che dovrà riuscire a scansare, il tutto allegrato dalle urla dei tifosi e da un'allegra musica.

Come gioco non è di difficile riuscita, basta solo metterci un po' di impegno. In schermo multicolor, cambiando i caratteri si disegna la pista con veduta aerea e se si riesce a far muovere lo schermo pixel per pixel viene fuori un bel gioco. Se poi ti interessa un gioco un po' più movimentato ho questa idea: il giocatore diventa una squadra di ladri che ha un pullmino bianco, il boss gli ha ordinato di andare in un paese di notte e far razzia di tutto quello che trovano. Essi hanno a disposizione dinamite, mitra, pistole etc. Per rifornirsi devono rientrare nella base mobile da cui sono arrivati e così via di nuovo fino a rubare il contenuto della banca, ma attenti alla polizia!».

Accidenti ragazzi! Qui si parla di Minigame (... senza offesa).

29 gennaio. Franco Orsogna

«Visto che realizzare un gioco di un unico genere non avrebbe soddisfatto la massa di smanettoni, ho pensato che la miglior cosa da fare sarebbe stata di realizzare un gioco con una mega-storia, con vari tipi di ambientazioni e generi (adventure, arcade, simulatori, spara e fuggi etc.) e integrarli in un unico gioco (come ad esempio SUMMER-GAMES della Epyx, che raggruppa in sé tutti i giochi estivi). Comunque questo gioco (et... voilà la proposta originale) dovrà partire dai primordi della storia, ovvero la preistoria e raggiungere il futuro (ad ogni periodo finito si passerà al successivo). Vi riassumo tutto nel seguente specchietto:

Periodo preistorico: un uomo primitivo che va a caccia e combatte dinosauri...

Periodo romano: invasione della penisola italica da parte dei Galli (il gioco consisterebbe nel respingerli o nel liberare un fortino dall'assedio)...

Periodo medievale: stufi degli arcade ci dedichiamo ad un impegnativo (e possibilmente grafico) adventure, ad esempio nelle lande dell'Inghilterra.

1800: colonizzazione di un nuovo territorio (più che altro un'esplorazione).

1900: simulatore di un biplano sullo stile «barone rosso»

2000: dopo una guerra atomica come sarà la vita? La proposta consiste in un gioco dove l'uomo deve sopravvivere.

3000: mille anni dopo la guerra nucleare, la terra è tornata a sorridere, ma questa volta i problemi provengono dall'esterno, cioè un'invasione UFO e tu dovrai...

Questa che vi ho proposto è solamente una bozza generale del MEGAGAME; si potrebbe proporre ai lettori di sostituire ad una data il tipo di gioco, oppure di inserire una nuova data e inventare il rispettivo gioco e così via.

Mi è venuto in mente un piccolo dubbio: quanto ci vorrà per realizzarlo? Penso che di tempo ce ne vorrà molto, comunque sono contento di aver dato un piccolo contributo alla vostra iniziativa».

Ecco quello che ci vuole. Tuttavia così è implementato male (e se a me gli adventure non piacciono?), ma siamo sulla buona strada.

2 febbraio. È la volta del primo critico (vedi MEGAPOSTA), Jacopo Piazzi

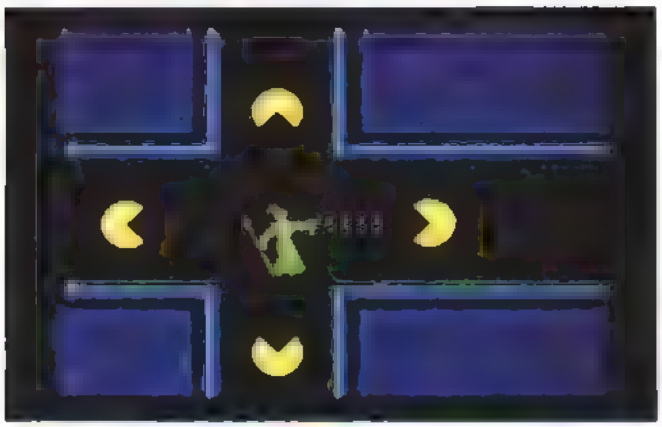
«A me piacerebbe moltissimo un gioco di guerra, ma non la solita astronave che deve far sempre le solite stragi interplanetarie; un bel gioco di strategia militare...»

E anche Jacopo ha detto la sua (per il resto ne riparliamo dopo).

Ancora 2 febbraio. Sergio di Fusco

«Più che un gioco, io proporrei un palinsesto elettronico. Adesso ti spiego. Supponi che la schermata di inizio rappresenti un foglio bianco, diviso in tanti quadretti, diciamo una trentina, e che su questi quadretti sia piazzato casualmente un piccolo segnalino che rappresenta il giocatore. Da quella casella egli è libero di muoversi nelle otto caselle adiacenti. Ma entrando nella casella, egli si troverà di fronte ad un videogame ridotto. Lì dovrà fare qualcosa, ma invece di raccogliere punti, come in tutti i giochi di questo mondo, egli muta colore allo sfondo, o alla cornice del gioco. Quando ha raggiunto il colore desiderato, può interrompere quel minigame particolare, e tornare allo schermo iniziale, dove la sua casella non è più bianca, ma del colore ottenuto tramite il gioco appena fatto. A questo punto egli potrà di nuovo muoversi verso una qualunque delle otto caselle circostanti, entrare in un nuovo minigame, e così via.

Scopo finale del gioco potrebbe essere quello di unire con una linea continua e dello stesso colore, due lati dello



schermo; cosa questa che potrebbe essere molto ardua ■ divertente se si gioca in due, usando ognuno un proprio colore. Giocando in solitario si potrebbe essere tentati a costruire una figura, un simbolo ■ un monogramma a vari colori, cosicché eliminando la top-list si potrebbe magari ottenere una piccola galleria di -quadri-, con sotto il nome dell'autore.

Per rendere il gioco vario, si potrebbe fare che, dal secondo turno in poi, i quadretti bianchi dello schermo iniziale cambino colore casualmente, lasciando invariati solo quelli già percorsi, corrispondentemente il colore dello sfondo di una partenza di ogni singolo minigame. In questo modo la difficoltà del gioco sarebbe di carattere dinamico ed ogni volta sempre vario, poiché per ottenere un certo colore potrebbero essere sufficienti a volte solo pochi -punti-, mentre in altre situazioni potrebbero esserne necessari parecchi.

Credo anche che sarebbe opportuno munire il gioco della possibilità di memorizzare la posizione raggiunta, per poter continuare in altro momento il compimento del -quadro-.

Ecco, questo sarebbe il gioco che mi piacerebbe giocare. Un gioco in cui entra il caso, l'estro ■ la riflessione. Un gioco che può durare quanto voglio io, una notte intera o solo un quarto d'ora. Un gioco che in realtà consente molti altri giochi.

Questa è solo un'idea, ed io non ho la benché minima possibilità ■ capacità di renderla un fatto. Spero quindi che ti possa interessare, così ci sarebbe qualcuno che la realizzerà per me.

I mini-giochi potrebbero anche essere di più, o anche di meno dei trenta proposti. Del resto, se lo chiedi tramite MC, credo ci saranno molti perfetti possessori dell'LM del C-64 pronti a spedirti i loro giochini.

Sarebbe un buon mezzo per fare del megagame 64 un videogioco tutto nostro, no?!».

Niente male questa idea....

3 febbraio. Alberto Bianchi

«Ho pensato che per fare un gioco un po' diverso dal solito bisognerebbe racchiudere tanti giochini in un videogame solo; mi spiego meglio: perché non fare il classico gioco della pallina che deve distruggere i mattoncini colorati (per intenderci tipo ARKANOID) con una variante? Ti faccio un esempio: sullo schermo ci sono dei mattoncini colorati ed altri numerati (magari dall'uno al dieci), ogni volta che la pallina colpirà un mattoncino numerato il game cambierà schermo ed apparirà un gioco differente (colpendo il numero uno si potrà giocare ad una mini battaglia spaziale, colpendo il numero due apparirà una mini corsa di automobili e così via...). Naturalmente ci sarà un limite di tempo per giocare ad ogni schermo ed alla fine del tempo prefissato il gioco tornerà allo schermo iniziale e cioè quello della pallina che deve colpire i mattoncini. Lo scopo del gioco è quello di raccogliere più punti possibili e di passare così ai livelli superiori».

Ancora lo stesso stile... (chissà...).

8 febbraio. Luca Persico

«Sono un appassionato di simulatori di volo e quindi mi piacerebbe che questo programma fosse appunto la simulazione del volo di un caccia a reazione ed in particolare che simulasse:
a) il decollo ■ l'atterraggio (ovviamente)
b) Combattimento aereo ed attacco al suolo.
c) Salvataggio con il paracadute (dopo il quale si riparte dall'aeroporto).
d) Possibilmente il rifornimento in volo ■ comunque il rifornimento di armi e carburante con l'atterraggio all'aeroporto.

Il videogioco che si deciderà di realizzare vorrei fosse tale che l'unico limite al tempo di gioco sia determinato dall'abilità del giocatore e quindi vorrei che non si verificassero situazioni tanto difficili da essere insormontabili (per es.: io ho dei videogiochi in cui ad un certo punto è impossibile rifornire di energia la propria astronave perché l'energia che viene data al precedente rifornimento è insufficiente a raggiungere il rifornimento successivo).

Un simulatore di volo eh? Ce ne son tanti però. Vediamo il prossimo lettore..

Sempre 8 febbraio. Vito Armenise

«A me piacciono (passando al dunque) i simulatori di volo ■ di guerra, i più fedeli possibili alla realtà.

Opero su diversi computer, ma a casa ho un C 128.

Tornando al gioco, la mia idea è quella di fare un simulatore ancora più simulatore degli altri. Mi spiego; molte volte nei quadri dei giochi o dei simulatori di aerei si vede l'aereo, quando su un vero aereo vedi le nuvole ■ basta. Molto probabilmente avrai visto WARGAMES; spero di sì perché la mia idea prende spunto da quel film. Immagina di lavorare in una base nucleare americana ■ arriva il segnale di allarme rosso perché dei missili russi si stanno dirigendo verso di noi.

L'ordine è di distruggere i missili e di contrattaccare. Così, alla guida di un caccia super moderno, si parte per la III guerra mondiale, e dopo aver distrutto i missili si parte alla volta della Russia per ricambiarli del favore.

Il resto è facile da immaginare, aerei russi, nuovi missili e così via. Anzi; si può cominciare il gioco da una postazione terrestre che deve difendersi dai missili e da diverse diavolerie russe, per

poi passare il controllo all'aereo. Un'altra idea sarebbe un simulatore (questo sempre) di un'astronave che parte dalla terra verso un falso buco nero (nel senso che è artificiale) perché questo sta per inghiottire il pianeta -verde- o -azzurro- come lo si voglia chiamare.

Il suo scopo è scoprire cosa c'è dietro questo mistero, e per far questo entra nel buco nero e sbuca (piaciuto il passaggio buco-sbuca?) in un altro universo e qui le sorprese saranno molteplici. Per esempio stelle rosse, pianeti triangolari o fatti di acqua e basta, e cose ancora più assurde che non sto qui a scriverti adesso perché, da come ho letto, per ora vuoi delle idee»

Ed ecco un altro simulatore .. basta!

Ultima lettera. (8 febbraio di nuovo). Cacao Maravigliao, pardon... Alberto Barbero

«La mia idea è quella di trasferire il gioco da tavolo — VII Legio —, della International Team su computer.

La settima legione è un immaginario corpo spaziale che si occupa di esplorare galassie sconosciute, per prendere contatto con altre razze aliene.

Un giocatore deve fare il ruolo di giudice e disegna la mappa della galassia segretamente.

Ogni giocatore allestisce la sua astronave e comunica la sua rotta al giudice, che sposta l'astronave corrispondente sulla mappa, e comunica l'avvistamento del pianeta, se la rotta seguita gli si avvicina.

Una volta che si scopre un pianeta, il giudice stabilisce le caratteristiche e se è abitato da esseri alieni.

Il giocatore a questo punto può scegliere se atterrare su di esso o proseguire il viaggio di esplorazione.

Se si atterra su di un pianeta si deve tentare una missione che può essere alleanza con gli abitanti, cattura di esemplari alieni rari, eliminazione di razze ostili e altro.

Se il giocatore riesce a portare a termine una missione, una volta tornato al pianeta di partenza, riceverà dei punti che accumulati gli permetteranno di essere promosso di grado. Vince chi per primo raggiunge il grado più alto.

Nella versione per computer si dovrà poter giocare in più persone, e tutto verrà gestito dalla macchina, facendo muovere a turno l'astronave di ogni giocatore secondo la rotta impostata dal giocatore stesso.

Il computer comunicherà al giocatore l'eventuale avvistamento di un pianeta e le sue caratteristiche.

Se il giocatore decide di atterrare sul pianeta, il gioco diventa un normale arcade a seconda del tipo di missione che si intende svolgere. Poi il giocatore, per vedersi assegnare i punti della missione, dovrà riportare la propria astronave al pianeta base.

Durante i viaggi di esplorazione possono accadere degli incidenti (ad es.: scontro con meteoriti, distorsioni che deviano la rotta senza avviso, astronavi ostili) che possono danneggiare la nave e costringere al rientro forzato al pianeta base»

Ma non avevo detto di inventare qualcosa di nuovo?

Bene bene, e così abbiamo finito. Tranquillizzo tutti coloro che non hanno visto pubblicata la loro lettera dicendo che probabilmente non è ancora arrivata e che sarà comunque presa in considerazione nella prossima puntata.

È il momento di giungere alle conclusioni. Veniamo subito al dunque. Sinceramente, tra tutte queste idee, solo una mi è particolarmente piaciuta e, se siete stati attenti ai brevi commenti espressi alla fine di ogni lettera, non dovreste incontrare difficoltà nell'individuarla. Parlo della proposta di Sergio di Fusco e cioè del suo megagioco/multigioco. Anche altri due lettori avevano pensato a qualcosa del genere (Franco Orsogna e Alberto Bianchi), ma non avevano centrato in pieno il bersaglio. Sergio ha proposto un gioco già di per sé divertente, che però, per essere giocato richiede l'abilità in tanti altri sottogiochi. È vero che anche Alberto aveva avuto un'idea molto simile, ma sinceramente di ARKANOID se ne sono visti un po' troppi. Franco invece, con la pensata di tanti megagiochi in stile diverso, ma collegati in modo forzato, non mi ha molto convinto. A questo punto credo che il giudizio spetti a voi (non voglio costringere nessuno). Scrivetemi ancora, ma questa volta non per proporre un nuovo megagioco, scegliete uno di questi da me considerato (o magari anche uno degli altri) oppure cominciate a pensare a trenta giochini richiesti da Sergio e a come migliorare la sua ottima proposta. Da parte mia credo che si possa concludere qui questa puntata, tuttavia devo ancora rispondere a delle richieste per la SCUOLA DI VIDEOGAME (che purtroppo, per mancanza di spazio, questo mese non faremo). Passiamo dunque alla ..

Megaposta

Jacopo Piazzi ci dice

«Caro Marco, sto leggendo le tue puntate sul megagame e vorrei farti una piccolissima critica: sono alla seconda lezione, ma mi sento in dovere di andare a ripetizione perché sinceramente io capisco poco di ciò che scrivi. Io credo che parecchia gente, come me, non ha sufficienti nozioni per comprendere ciò che dici. Per esempio: pagina 179, prima colonna, tredicesima riga partendo dal basso, dove dice: — .. se qualcosa non è stata visualizzata dall'inizio perché visualizzarla in parte? Bella volpata!.. —. Bene queste e altre cose non mi sono affatto chiare perciò ti pregherei di essere un po' più esplicito».

Probabilmente se specificavi meglio cosa non ti era chiaro potevo darti qualche delucidazione. Comunque cercherò di non essere troppo sibillino in seguito (nella frase incriminata stavo solo prendendo in giro un difetto del VIC II).

Luca Persico ci chiede

«Per quanto riguarda la SCUOLA DI VIDEOGAME vorrei che tu spiegassi come è possibile in un simulatore di volo realizzare in modo rapido la rotazione dell'orizzonte (e di tutto quanto sta a terra e in cielo) durante una virata, ed in genere come si può modificare velocemente la scena esterna alla cabina in conseguenza dei movimenti dell'aereo (conosco un simulatore di volo di un caccia a reazione che lo fa in modo molto rapido). Inoltre ho un altro problema da sottoporti, al quale ne io ne coloro ai quali mi sono rivolto siamo riusciti a trovare una soluzione (nonostante numerosi tentativi): nei game in cui due giocatori si fronteggiano capita a volte che entrambi i giocatori devono usare la tastiera, il computer, però, legge un solo tasto alla volta, cioè se ne vengono premuti contemporaneamente due o più il computer ne legge uno solo e gli altri li ignora del tutto. Io vorrei che tu spiegassi il modo per poterli leggere...».

Tutto quello che chiedi è da trattare in modo accurato e quindi ne riparleremo nelle prossime puntate alla SCUOLA DI VIDEOGAME..

L'Italian Team chiede

«... Tu sai come si fa a cambiare i caratteri sul 128?»

Chiederemo in giro...

Fine della posta e fine della puntata. A presto.

di Marco Pesce

Un videogioco tutto nostro

quinta puntata

La volta scorsa abbiamo decisamente esagerato. Tutte lettere e niente arrosto, o meglio, niente Scuola di Videogame. È addirittura arrivato un rimprovero dal direttore («mi raccomando, non lo fare più»). D'accordo sono pentito. Quello che più mi fa tremare ■ che certamente, tra non molto, sarò sommerso da una valanga di lettere di protesta, inviate da tutti quei lettori che non hanno digerito il polpettone

Un marchio per il nostro megagame

Naturalmente stavo scherzando. Sono convito che il numero di marzo sarà quello più seguito (già, «sarà»; quando scrivo ancora non è uscito). È stata una trovata senza dubbio geniale, per aumentare le vend... cioè per esporre a tutti i lettori il... «parere dei lettori» (che bufalata questa... CUT CUT CUT).

Ripristiniamo un atteggiamento serio. Dunque sono arrivate altre lettere e quindi altre proposte. Tra queste l'unica degna di interesse è stata quella del lettore Emilio Orione. In sostanza questi proporrebbe di realizzare un gioco in cui un personaggio di nome MUFFY deve destreggiarsi in una città, al fine di diventare il capo banda del suo gruppo di amici. Il «destreggiarsi» consiste nel dimostrare la propria abilità in varie «discipline», che naturalmente ■ identificano in vari sottogiochi. L'idea di Emilio è da lui stesso definita «adatta ad un'implementazione su rivista» in quanto permette di realizzare tutti i giochi che vogliamo e quindi accontentare qualsiasi giocatore.

A questo punto credo che non ci siano più dubbi; il gioco da realizzare è un multi-game. Il problema ora ■ quale scegliere tra i quattro ■ nostra disposizione (ricordate i tre giochi da me evidenziati nella scorsa puntata?). Sinceramente, per quanto mi riguarda, il problema l'avrei risolto, ovvero sarei favorevole alla realizzazione del megagioco di Sergio di Fusco che mi sembra quello più originale e, perché no, più divertente.

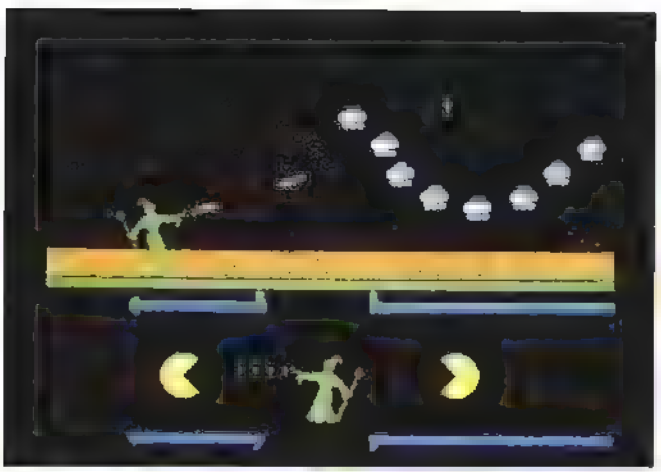
Qualcuno mi ha accusato di essere troppo lento (e ha ragione). Dobbiamo darci una mossa. Io comincerei ■ fare qualcosa, o meglio, inizierei con un'impostazione generale del gioco. Se poi il mese prossimo si scopre che «qualcuno» non ■ perfettamente d'accordo con la mia scelta, vorrà dire che ricominceremo da capo.

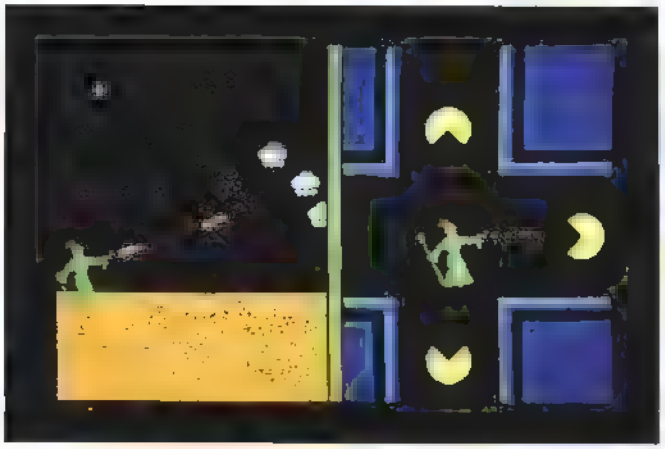
Dunque, riassumiamo brevemente il gioco di Sergio. Si tratta, come detto, di un multigioco. Per accedere ■ vari giochi in esso contenuti occorre semplicemente eseguire una scelta, dopo la quale dovremo dimostrare la nostra abilità al fine di alterare il colore del bordo dello schermo (!). Questo è necessario in quanto la scelta ■ effettua su un riquadro suddiviso in tanti quadretti bianchi. Spostando quindi un nostro segnalino su uno di questi quadretti, decidiamo automaticamente di cambiarne il colore. Durante il gioco, infatti, il bordo dello schermo corrisponde al colore del quadratino. In qualunque momento possiamo smettere la partita. Torneremo al riquadro iniziale e noteremo che il quadratino ha cambiato colore. Potremo allora effettuare una nuova scelta. Lo scopo del gioco potrebbe essere quello di realizzare una «pittura artistica» oppure di riprodurre una figura prestabilita o comunque di «imbrattare» la tela, cioè il riquadro. Il gioco ■ presta bene anche a una partita che coinvolge due giocatori. Per esempio, supponiamo che ognuno debba a tutti i costi generare un proprio colore in modo da farlo predominare rispetto a quello dell'avversario. Con l'aggiunta di questa opzione il gioco sarebbe oltretutto vantaggioso per il risparmio dell'impianto di riscaldamento del vostro stabile, ottenuto inevitabilmente grazie all'aumento della temperatura dovuta all'attrito procurato sui due joystick (forse ho esagerato un pochino). A parte gli scherzi, abbiamo fra le mani un gioco che potenzialmente può risultare tra i migliori mai visti sul C64 (e questa volta non esagero). Chiaramente, come ho più volte ripetuto, bisogna impegnarsi, ma non è il momento di far prediche, vediamo invece di passare veramente all'azione. Innanzi tutto, siamo sicuri di sapere esattamente cosa vogliamo? No, ma per il momento non importa. Diamo un'impostazione generale al programma. Ci occorre una routine fondamentale in grado di:

1) generare il riquadro di scelta ■ quindi controllare la scelta,
2) lasciare il controllo a uno dei giochi,
3) riprendersi il controllo e aggiornare il riquadro di scelta.

Queste sono chiaramente solo alcune delle funzioni principali che la MAIN-ROUTINE (d'ora in poi la chiameremo così) dovrà svolgere. Tuttavia sono quelle che dovrà assolutamente svolgere.

A questo punto occorre specificare





Il doppio schermo così o così?

altri dettagli del gioco. Supponiamo di aver deciso che giocando da soli contro il computer bisogna riprodurre una determinata figura o, se vogliamo, un determinato «quadro». È chiaro che il quadro (senza virgolette) originale sarà generato dal computer per poi poter essere osservato dal giocatore. Mi viene in mente proprio ora una piccola variante da aggiungere al gioco. Potremo fare in modo che il quadro venga visualizzato una sola volta all'inizio ■ poi a «pagamento» durante il resto della partita. La quota potrebbe essere il subire una piccola menomazione al nostro elaborando . (come sono cattivo) Dando un'occhiata più accurata alla versione a due giocatori mi rendo conto che dovrà essere impostata in maniera notevolmente diversa, quasi fosse un altro gioco. Infatti se è vero che i due giocatori debbono giocare contemporaneamente, è anche vero che occorre suddividere lo schermo in due parti e quindi avere il classico doppio schermo già visto in molti altri giochi. Il doppio schermo dovrà persistere sempre, perché anche il riquadro di scelta deve avere un accesso indipendente per i due giocatori, cioè in qualunque momento ogni giocatore deve potervi accedere senza disturbare il «lavoro» dell'avversario. Pensandoci bene però, così facendo bisognerebbe disegnare i vari giochi per due formati di schermo, ovvero per l'opzione a un giocatore (schermo pieno) e quindi per l'opzione a due (schermo «mezzo + mezzo»). Questo sarebbe assolutamente dispendioso in quanto a memoria. D'altronde una routine adibita alla riformattazione delle dimensioni del campo di gioco (a seconda se giochiamo da soli o in due) mi sembra un'impresa da folli; quindi, facendo un esame obiettivo della situazione, dobbiamo giungere ad un piccolo compromesso. Vada pure per il formato diverso dei riquadri-scelta, ma per quanto riguarda i giochi... se si gioca da soli avremo una parte dello schermo «vuota» (credevate di peggio vero?). Abbiamo fatto quindi un altro passo avanti; ci necessita una routine che genera il quadro da riprodurre, una che lo visualizza (gratis o a pagamento) e una che provoca il danneggiamento della pittura. Inoltre la MAIN-ROUTINE deve prevedere la partita a due giocatori e in tal caso occorre modificarla radicalmente perché deve:

1) generare i due riquadri di scelta;
2) gestire le scelte dei due giocatori, in modo indipendente;
3) cedere il controllo ■ uno o due giochi che dovranno «girare» anche contemporaneamente;
4) in caso di ritorno al riquadro principale, aggiornare la configurazione dello stesso;
5) controllare la vincita di uno dei due giocatori.

A proposito di vincite; nella versione a giocatore singolo occorre anche la routine che si cura di controllare l'esatta riproduzione e quindi che agisce di conseguenza, «avvisando» il giocatore. Tenendo conto di questo e delle altre aggiunte fatte prima, aggiorniamo la MAIN-ROUTINE versione un giocatore. Dunque, oltre ai punti 1) 2) e 3) già elencati, essa deve:

4) visualizzare a pagamento il quadro originale ■ danneggiare, di conseguenza, quello in riproduzione;
5) controllarne l'esatta riproduzione e, se verificata, avvisare il giocatore.

Occorre inoltre inserire, prima del punto 1), la routine che genera ■ quadro da riprodurre e che lo visualizza per una sola volta.

Per quanto riguarda la vincita della versione a due giocatori, possiamo fare in modo che ■ verifichi quando uno dei due giocatori riesce a disegnare una determinata figura. A prima vista potrebbe sembrare uno scopo banale perché troppo simile a quello da svolgere giocando da soli, ma riuscite ad immaginarvi la competizione che può generarsi tra i due contendenti mentre disegnano sullo stesso spazio e si rovinano ■ vicenda il loro quadro? Io sì. Chiaramente la figura da «ricopiare» in questo caso sarà più piccola ■ quindi meno impegnativa, ma quando si gioca in due il divertimento è tutto nell'esclamare: «Ho vinto io!».

Dando un seguito a quanto detto, dobbiamo aggiungere anche alla versione TWO-PLAYER (scusate «l'inglesismo», ma non posso ripetere per 200 volte la stessa parola) ■ routine «visualizza ■ pagamento» e «controlla l'esatta riproduzione», più quella generatrice delle due figure da ridisegnare.

Agli scettici che dicono «se continuate di questo passo non finirete mai» rispondo che con questa puntata abbiamo fatto già un bel passo avanti (...fidatevi).

Il prossimo mese arriveranno le vostre opinioni sulla mia scelta (cfr. la scorsa puntata) e nel frattempo vi faccio la seguente domanda: perché non cominciate a mandarmi una dimostrazione delle vostre capacità? Mi spiego meglio: perché non mi inviate un giochino di vostra produzione, magari che non siete riusciti a completare? Se degno di nota, cercherò di inserirlo nel MEGAGAME apportandovi le dovute modifiche (e mi rivolgo anche a quei programmatori che vedono con un occhio di invidia questa iniziativa). Tra l'altro occore anche pensare alle parti di contorno del gioco e cioè ■ tutte quelle finezze che lo renderanno un gioco di classe, quindi presentazione, particolari effetti o accorgimenti grafici ■ via dicendo. Anche per far questo non occorre essere programmatori, quindi datevi da fare.

Scuola di videogame
Moltiplicazione degli sprite

Eccoci di ritorno dopo il salto del numero di marzo. Questo mese vorrei cominciare a parlare di LM e quindi distruggere quell'orribile maldicenza che lo classifica come il piu complicato e, al tempo stesso, stupido dei linguaggi. In questa nostra spiegazione cercheremo, per quanto possibile, di fare analogie con il Basic, che, a quanto pare, è il linguaggio che conoscete un po' tutti (a mio avviso, solo perché fornito in dotazione con la macchina, con tanto di «istruzioni per l'uso»).

Nella sua accezione piu classica, il linguaggio macchina e il vero e unico linguaggio che il computer capisce in modo diretto, cioè senza l'aiuto di un interprete. Vi ricordo che il sistema operativo non è l'interprete del linguaggio macchina, ma solo una serie di subroutine che ne facilitano l'uso. Tutte quelle che vi dirò non saranno parole buttate al vento, ma serviranno per la comprensione del listato proposto che, come si intuisce dal titolo, serve ad aumentare gli sprite disponibili sul C64.

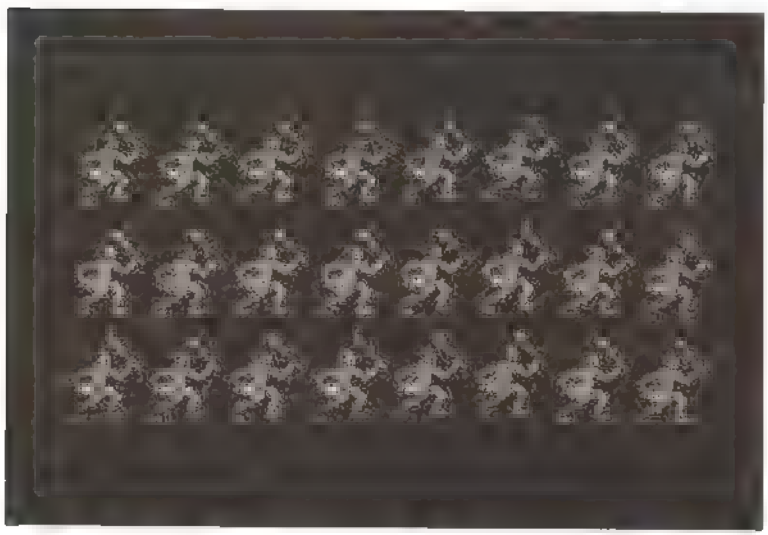
Piccola premessa: seguendo la filosofia di questa rubrica, tenterò di non annoiare nessuno e quindi di non dilungarmi troppo sulle cose semplici e, nello stesso tempo, di non parlare «troppo difficile».

Dunque, come molti di voi già sanno, il C64 è un computer dotato di:

64 KByte di RAM
20 KByte di ROM
4 KByte di I/O

Il microprocessore è il circuito centrale che praticamente comanda tutto il resto. Per accedere agli altri circuiti (generatore del suono, generatore della grafica etc.) deve in qualche modo essere collegato con questi. Detto collegamento avviene tramite i 4K di I/O (Input/Output). Per comprendere quanto detto occorre prima dare un'occhiata al funzionamento del microprocessore stesso. Questi è in grado di «indirizzare» 64 KByte di locazioni di memoria, cioè puo accedere (in scrittura o in lettura) a un numero limitato di byte. Consideriamo quello che succede all'accensione del Commodore 64; ci ritroviamo in uno stato di pieno disordine. Il microprocessore ha il compito di riordinare tutto. Partendo da una locazione di memoria forzata, comincia la sua opera, «prendendone» il contenuto. Questo vuol dire che un byte della ROM, che contiene il sistema operativo (S.O.), viene memorizzato all'interno del microprocessore. Cosa se ne fa di questo byte? Semplice: lo traduce in una istruzione da eseguire. A seconda del tipo di istruzione il microprocessore puo prendere anche i successivi byte (uno o due) e considerarli parte integrante della stessa. Questo perché un'istruzione non è sempre definita da un solo byte. In ogni caso i successivi (sempre uno o due) byte non sono necessari per la decodifica dell'istruzione, ma servono, ad esempio per fornire dei dati da utilizzare unitamente con essa.

Supponiamo quindi che la nostra prima istruzione sia stata interpretata e che gli eventuali dati siano stati prelevati dalla ROM. Il microprocessore, a questo punto, esegue l'istruzione. Questa si identifica normalmente in un'operazione che modifica lo stato interno del microprocessore o che modifica una locazione di memoria. Visto che siamo alla partenza e che quindi dobbiamo riordinare lo stato di tutti i circuiti del computer, questa prima ed altre saranno adibite a tale scopo. Quindi il microprocessore continuerà con il prelievo delle istruzioni (e con la relativa esecuzione) fin quando tutto sarà rimesso in ordine. Come fa il microprocessore a comandare i circuiti che lo circondano? Anche qui la risposta è molto semplice: i 4 KByte di I/O, che avevamo descritto prima, non sono altro che delle locazioni



: 24 sprite

di memoria «RAM» interne a detti circuiti (quindi non 4K di RAM normale). Scrivendo dentro questa particolare RAM il microprocessore automaticamente impone un comando a questi benedetti circuiti che reagiscono di conseguenza. I quattro kappa di RAM sono posizionati a partire dalla locazione 53248 (D000 in esadecimale). A questo punto è d'obbligo porsi la seguente domanda: «ma se il microprocessore può indirizzare solo 64 KByte, come fanno ad esserci 64K di RAM, 20K di ROM e 4K di I/O?». Mi auguro che detta domanda ve l'eravate già posta da molto prima. La risposta comunque arriva solo adesso. Con opportuni artifici, il C64 è in grado di scegliere se indirizzare o no una parte di RAM o di ROM quindi l'input/output. Deve cioè fare in modo che il totale della «roba» da indirizzare sia di 64 KByte. Tutto qui. Normalmente il nostro Commodore predispone:

44 KByte di RAM
16 KByte di ROM
4 KByte di I/O

Le ROM comprendono, come noto, il sistema operativo e il Basic, mentre i 4K «scartati» contengono la forma dei 512 caratteri grafici disponibili sul 64. Dato che non è il microprocessore quello che vuole saperne il contenuto (si tratta invece del VIC-II, Video-Chip-Interface), sono eliminati e ci permettono perciò di usare altri 4K di RAM. Intendiamoci, in qualunque momento la mappa di memoria accessibile può essere alterata (volontariamente) quindi non c'è assolutamente da preoccuparsi per la restante RAM non utilizzata. In tal caso, però, il prezzo da pagare è la momentanea impossibilità di accedere alla ROM e all'input/output fino a configurazione originale ripristinata. Confusi? Vi concedo 5 minuti d'aria... poi riprendremo di corsa...

...Come vola il tempo eh?

Bene, chiarite queste noiose, ma non trascurabili, cosette, possiamo andare avanti con la lezione di linguaggio macchina, descrivendo le caratteristiche del li-

```
    PC  SR AC XR YR SP
 E37E EA31 A4 F1 30 11 D6
.7000 78       SEI
.7001 A9 7F    LDA #$7F
.7003 8D 0D DC STA $DC0D
.7006 A9 30    LDA #$30
.7008 A2 70    LDX #$70
.700A 8D 14 03 STA $0314
.700D 8E 15 03 STX $0315
.7010 EA       NOP
.7011 A9 00    LDA #$00
.7013 8D 12 D0 STA $D012
.7016 AD 11 D0 LDA $D011
.7019 29 7F    AND #$7F
.701B 8D 11 D0 STA $D011
.701E A9 01    LDA #$01
.7020 8D 1A D0 STA $D01A
.7023 58       CLI
.7024 60       RTS
.7025 EA       NOP
.7026 EA       NOP
.7027 EA       NOP
.7028 EA       NOP
.7029 EA       NOP
.702A EA       NOP
.702B EA       NOP
.702C EA       NOP
.702D EA       NOP
.702E EA       NOP
.702F EA       NOP
.7030 A9 FF    LDA #$FF
.7032 8D 19 D0 STA $D019
.7035 A9 60    LDA #$60
.7037 A0 70    LDY #$70
.7039 8D 14 03 STA $0314
.703C 8C 15 03 STY $0315
.703F A9 60    LDA #$60
.7041 8D 12 D0 STA $D012
.7044 AD 11 D0 LDA $D011
.7047 29 7F    AND #$7F
.7049 8D 11 D0 STA $D011
.704C A9 00    LDA #$00
.704E 8D 20 D0 STA $D020
.7051 20 00 70 JSR $7000
.7054 4C BC FE JMP $FEBC
.7057 EA       NOP
.7058 EA       NOP
.7059 EA       NOP
.705A EA       NOP
.705B EA       NOP
.705C EA       NOP
```

Il listato del «Moltiplicatore di sprite»

```
.705D EA       NOP
.705E EA       NOP
.705F EA       NOP
.7060 A9 FF    LDA #$FF
.7062 8D 19 D0 STA $D019
.7065 A9 90    LDA #$90
.7067 A0 70    LDY #$70
.7069 8D 14 03 STA $0314
.706C 8C 15 03 STY $0315
.706F A9 90    LDA #$90
.7071 8D 12 D0 STA $D012
.7074 AD 11 D0 LDA $D011
.7077 29 7F    AND #$7F
.7079 8D 11 D0 STA $D011
.707C A9 01    LDA #$01
.707E 8D 20 D0 STA $D020
.7081 20 C0 70 JSR $70C0
.7084 4C BC FE JMP $FEBC
.7087 EA       NOP
.7088 EA       NOP
.7089 EA       NOP
.708A EA       NOP
.708B EA       NOP
.708C EA       NOP
.708D EA       NOP
.708E EA       NOP
.708F EA       NOP
.7090 A9 FF    LDA #$FF
.7092 8D 19 D0 STA $D019
.7095 A9 30    LDA #$30
.7097 A0 70    LDY #$70
.7099 8D 14 03 STA $0314
.709C 8C 15 03 STY $0315
.709F A9 C0    LDA #$C0
.70A1 8D 12 D0 STA $D012
.70A4 AD 11 D0 LDA $D011
.70A7 29 7F    AND #$7F
.70A9 8D 11 D0 STA $D011
.70AC A9 02    LDA #$02
.70AE 8D 20 D0 STA $D020
.70B1 20 E0 70 JSR $70E0
.70B4 4C 31 EA JMP $EA31
.70B7 EA       NOP
.70B8 EA       NOP
.70B9 EA       NOP
.70BA EA       NOP
.70BB EA       NOP
.70BC EA       NOP
.70BD EA       NOP
.70BE EA       NOP
.70BF EA       NOP
.70C0 A2 10    LDX #$10
.70C2 BD 00 71 LDA $7100,X
```

```
.70C5 9D 00 D0 STA $D000,X
.70C8 CA       DEX
.70C9 10 F7    BPL $70C2
.70CB 60       RTS
.70CC EA       NOP
.70CD EA       NOP
.70CE EA       NOP
.70CF EA       NOP
.70D0 A2 10    LDX #$10
.70D2 BD 20 71 LDA $7120,X
.70D5 9D 00 D0 STA $D000,X
.70D8 CA       DEX
.70D9 10 F7    BPL $70D2
.70DB 60       RTS
.70DC EA       NOP
.70DD EA       NOP
.70DE EA       NOP
.70DF EA       NOP
.70E0 A2 10    LDX #$10
.70E2 BD 40 71 LDA $7140,X
.70E5 9D 00 D0 STA $D000,X
.70E8 CA       DEX
.70E9 10 F7    BPL $70E2
.70EB 60       RTS
.70EC EA       NOP
.70ED EA       NOP
.70EE EA       NOP
.70EF EA       NOP
.70F0 EA       NOP
.70F1 EA       NOP
.70F2 EA       NOP
.70F3 EA       NOP
.70F4 EA       NOP
.70F5 EA       NOP
.70F6 EA       NOP
.70F7 EA       NOP
.70F8 EA       NOP
.70F9 EA       NOP
.70FA EA       NOP
.70FB EA       NOP
.70FC EA       NOP
.70FD EA       NOP
.70FE EA       NOP
.70FF EA       NOP
```

stato proposto. Esso è in grado di visualizzare contemporaneamente ben 24 sprite, ma, apportandovi alcune modifiche, questo numero può ancora incrementare.

Per visualizzare 24 sprite (3 X 8) occorre usare la tecnica dell'interruzione (IRQ). Nel nostro caso si tratta di interruzioni video, cioè che avvengono quando il pennello elettronico raggiunge una determinata posizione. Detta posizione è impostabile da parte del programmatore e, soprattutto, non deve essere per forza una sola. In altre parole, stiamo mettendo in pratica quanto visto nel numero di febbraio a proposito della «riutilizzazione degli sprite» durante la stessa «pennellata» video. Il listato, al contrario di quanto potrebbe sembrare a prima vista, è tutt'altro che complicato. Richiede comunque la conoscenza di alcune istruzioni che ci apprestiamo a descrivere. A proposito, non ditemi che non avete a portata di mano un Monitor-Assemblatore. È assolutamente indispensabile per programmare in linguaggio macchina «come si deve». Per chi non lo sa, un Monitor-Assemblatore è il programma che facilita il compito di trascrittura delle istruzioni nel computer e che oltretutto permette di effettuare alcune operazioni utili come il riempire con un determinato valore una grande area memoria molto velocemente o copiare un blocco di RAM/ROM a partire da una locazione differente da quella attuale etc. Quindi se siete seriamente intenzionati a diventare dei programmatori, occorre procurarselo.

Tornando a noi, vediamo innanzi tutto di capire come si effettua la lettura di un disassemblato come quello del nostro listato. Partendo dall'alto a sinistra vediamo, in successione, le diciture:

```
PC IRQ SR AC XR YR SP
```

Sotto ognuna di queste sigle troviamo i valori.

```
E37B EA31 A4 F1 30 11 DC
```

I nomi elencati prima sono nomi attribuiti a dei registri interni (RAM interna) al microprocessore, tranne l'IRQ, che è invece il «vettore» di salto che indica la routine da eseguire in caso di IRQ (quella normalmente provocata dal timer ogni sessantesimo di secondo). Diamo ora una breve spiegazione alla funzione di questi registri. Il PC (o Program Counter) è un registro a 16 bit (2 byte) che contiene un valore corrispondente all'indirizzo dell'istruzione da prelevare dalla memoria. L'SR (o Status Register) contiene dei valori che servono a indicare alcune condizioni generatesi in seguito all'esecuzione delle istruzioni precedenti (non preoccupatevi, ne riparleremo). L'AC (Accumulatore) è il registro nel quale vengono depositati tutti i dati (a 8 bit) che devono subire un'operazione e quindi una modifica. L'XR (Registro X) viene normalmente utilizzato in operazioni che richiedono un contatore (ma non è un contatore) o un puntatore; è a 8 bit e non supporta tutte le operazioni possibili, come fa l'Accumulatore. L'YR (Registro Y), come è facile intuire, è praticamente uguale a l'XR (cioè può fare quasi le stesse cose).

Infine l'SP (Stack Pointer) che è un puntatore a una particolare zona di RAM, adibita al momentaneo deposito di dati. Che fatica! Dunque, i valori (sono in esadecimale) di cui parlavamo, corrispondono al contenuto di detti registri e sono relativi all'ultima volta che il microprocessore ha eseguito un programma (ovviamente prima dell'avviamento del Monitor-Assemblatore, che è anch'esso un programma e quindi li modificherebbe in continuazione).

Queste due righe vengono visualizzate non appena il M.-A. viene fatto «partire» e possono essere visualizzate, normalmente, premendo il tasto R seguito da un return. Ci siamo dilungati un po' troppo vero? Proseguiamo. La terza riga comincia a rappresentare il programma vero e proprio, e meglio il disassemblato di questo. Il primo numero (sempre in esadecimale), cioè il 7000, è la prima locazione di memoria occupata da questi. Il numero che segue (78) rappresenta l'equivalente in esadecimale di detta locazione ed è un'istruzione.
La scritta «SEI» è il codice mnemonico dell'istruzione stessa, ovvero un nome che serve per ricordare più facilmente quell'istruzione (vi immaginate «che macello» se dovessimo ricordare a memoria il numero corrispondente a ciascuna istruzione?). Per scrivere questa istruzione in memoria io ho informato il mio M.-A. che volevo scrivere un programma a partire dalla locazione $7000 (il $ sta per esadecimale), con la seguente istruzione:

```
A7000
```

Di seguito ho scritto, sulla stessa linea, il codice «SEI» e quindi ho premuto il tasto return. Automaticamente il M.-A. ha assemblato il mio codice scrivendo al posto della mia riga, quella seguente:

```
A7000 78 SEI
```

e ne ha aggiunta un'altra come questa qui sotto:

```
7001
```

Dopo il 7001 io ho scritto il codice LDA #$7F e così via. Non vi credo degli handicappati; penso solo che così è più semplice capire la lezione!

Continuiamo. Se arrivati a un certo punto ci vogliamo fermare, basta premere return a vuoto, ovvero senza scrivere nessuna istruzione. Possiamo così «listare» il nostro codice in Assembly, imponendo la seguente riga e premendo return:

```
D7000
```

La lettera D sta per «Dissassembla», così come la lettera A sta per «Assembla».

Finalmente possiamo spiegare cosa significano tutte quelle istruzioni. L'istruzione «SEI» fa in modo, ovviamente quando viene eseguita, che tutte le IRQ vengano disabilitate (niente più IRQ). Per riabilitarle occorre l'istruzione «CLI». Dimenticavo che per far partire un programma occorre impostare:

```
G7000 (ma quando è finito, non ora!)
```

oppure, se siamo in ambiente Basic il classico SYS seguito dall'indirizzo iniziale (cioè SYS 28672).

L'istruzione «LDA» ha diversi formati, ma nel nostro caso viene usata per «caricare» nell'accumulatore il valore $7F (il # vuol dire che si tratta di un dato e non di una locazione di memoria). Questo è il primo esempio di istruzione che occupa 2 byte. Tornando al programma, se viene eseguito solo fino a questo punto, esso disabilita le IRQ e mette nell'accumulatore il valore $7F, la successiva istruzione serve a «scaricare» l'accumulatore e anche essa ha diversi formati, ma nel nostro caso deposita detto valore nella locazione di memoria $DC0D (è uno dei registri dell'I/O). Cosa abbiamo fatto finora? Più semplice che mai. Abbiamo messo nella locazione $DC0D il valore $7F. Questo ci servirà per il regolare svolgimento delle IRQ.

Mi rifiuto di spiegarvi la funzione delle successive istruzioni contenute nelle locazioni antecedenti la $7010. Servono comunque a depositare i valore $30 e $70 rispettivamente nelle locazioni $0314 e $0315. Queste due ultime locazioni contengono il byte basso e il byte alto del vettore di IRQ e noi così facendo lo abbiamo modificato. Quindi alla prossima IRQ il microprocessore eseguirà il codice contenuto a partire dalla locazione $7030. Lo sapevo che sarebbe andata a finire così... adesso si arrabbieranno tutti... ma... non c'è più spazio. Vi prego, siate comprensivi continueremo la prossima puntata. Nel frattempo studiatevi (se ci riuscite) il listato e cercate di capirne il funzionamento... Vi ringrazio e vi lascio alla Megaposta.

Megaposta

Non arrabbiatevi!!!

Ho seguito con interesse la rubrica Megagame 64 fino dalla prima puntata e devo dire che avete davvero avuto una buona idea. Non fate caso a qualche invidioso che dice di lasciar perdere perché è impossibile creare un gioco per posta e non fate caso ugualmente a qualche altro presuntuoso che, dopo aver fatto il saputello con uno sfoggio di cultura non richiesto, suggerisce egoisticamente di «passare attraverso MC-Link». Mi ha sorpreso il caro Gianni Z. che si pensa di essere Einstein proponendo il quiz delle tre età dei figli. Probabilmente la risposta è una freddura, perché con quei tre dati e matematicamente impossibile dare una risposta. Ma perché date importanza a gente come Gianni Z. facendo occupare alla sua suppongo chilometrica lettera ben 66 righe? Certo l'idea di un multigioco nel tempo è originale, ma è solo questa la cosa originale

Alessandro Merolli, Roma

È chiaro che ognuno è libero di esprimere la propria opinione... Alessandro ha detto la sua. Tuttavia «hai toppato» prendendotela con Gianni Z. e con il suo giochino che, come puoi constatare dalla «lettera» del prossimo lettore, ha una soluzione rigorosamente matematica. Questo comunque non vuol dire che chi propone indovinelli ha sempre ragione...

Soluzione...

Caro Marco, ti scrivo per rispondere al quesito proposto da Gianni Z. nella rubrica Megaposta. Mi pare sia molto semplice e non ci ho messo molto per risolverlo. Ma veniamo al dunque:
— le età sono tre ed il loro prodotto fa 36, ci viene detto: poiché si assume che le età siano intere, bisogna scomporre in fattori il prodotto. 1-2-2-3-3, raggruppando i fattori in modo da ottenere tre numeri, si ottengono le seguenti otto combinazioni (escludendo le duplicazioni):

1-1-36
1-2-18
1-3-12
1-4-9
1-6-6
2-2-9
2-3-6
3-3-4

— con la seconda risposta ci viene detto che la somma delle tre età è pari al numero civico della casa di fronte, ma l'amico risponde che l'indizio non è ancora sufficiente; le varie combinazioni indicate danno come somma rispettivamente:

38-21-16-14-13-13-11-10

Bisogna escludere le somme che danno luogo ad una sola soluzione, perché in questo caso l'amico non avrebbe bisogno (pur vedendo il numero civico) di una terza domanda: la somma è quindi 13! Si tratta ora di scegliere tra 1-6-6 e 2-2-9;
— con l'ultima risposta ci viene detto che il più piccolo dei figli ha gli occhi azzurri: nel caso 2-2-9 il più piccolo NON ESISTE dato che ci sono due gemelli di due anni; la soluzione cercata è quindi:

1-6-6 !!!

Vinicio Coletti, (inviata tramite MC-Link)

Così, grazie a Vinicio, abbiamo arrestato la caduta dei capelli di tutti quelli che si stavano scervellando per trovare la soluzione.

Strane proposte

...Proporrei di fare il gioco in Basic semplice (mi spiego: senza «Simon's Basic» o linguaggio macchina), per fare in modo che veramente tutti possano provare l'emozione di farsi un vero gioco di quelli tosti. Quando spieghi però (il «però» è onnipresente), dovresti essere un po' più «ordinato», cioè, ogni tanto non riesco a seguire il discorso, ma forse succede perché sono un po' «paraplegico». Non so se questa lettera verrà pubblicata, ma se succede, potrò festeggiare perché sarebbe la prima volta che il mio nome (non spaventarti per la sua lunghezza) compare su una rivista.
Christian Kokkinomagoulos, Bologna

Aiuto... chiamate i pompieri!
A parte il nome che mi sembra abbastanza comune (!), ma ti rendi conto di quello che hai scritto? Cercare di realizzare un Megagame in Basic è come provare a scavare una galleria sotto al Monte Bianco con paletta e secchiello da spiaggia! Prova a seguire le mie lezioni e vedrai che non è poi così difficile «sto' linguaggio macchina».

Colonna sonora

Salve Marco, mi chiamo Gianfranco Gramiglia, ho 21 anni e abito a Napoli...
... A cosa ti posso servire?
Interessi principali della mia umile esistenza:

PRIMO-la mia ragazza
SECONDO-pensare
TERZO-suonare e digitare (pari merito)
ecc. ecc

Il primo non può interessarti (anzi, non deve).
Il secondo può interessarti relativamente visto che potrei mettermi a pensare a un nuovo videogame.
Il terzo: digitare (programmare) non sono un mostro (forse con l'aiuto di MC e dei tuoi articoli chissà...), decisamente rimane suonare. Chiedi una colonna sonora e sarà fatta; veloce, avvincente, dolce, romantica, lenta, maniacale, hard, disco, come vuoi tu e io la farò (ehm, tenterò, ma tenterò fino allo spasmo visto che ho i mezzi).
P.S. Il «nostro» videogame hai detto deve sfondare, ma dove? Italia, Europa, U.S.A., per quale età dev'essere? Dovresti imporre dei limiti.

Gianfranco Gramiglia, Napoli

Ragazzi abbiamo trovato il nostro musicista! Caro Gianfranco, finora sei stato l'unico che ha pensato alla musica... mandami una musicassetta con un tuo elaborato... vediamo quello che si può fare! Comunque non capisco perché vuoi imporre limiti...

Referendum

...Purtroppo il mio apporto alla rubrica potrà essere, credo, molto piccolo per via della scelta del computer su cui scrivere il gioco, il 64. Sono un felice possessore di Amiga 1000, proveniente tra l'altro da uno Spectrum 48k, quindi conosco ben poco il mio fratellino minore. Capisco che aprire il campo un po' a tutti i computer avrebbe comportato una mole di lavoro troppo grossa, ma almeno un Referendum si poteva fare. Dopo tutto il 64 è il più diffuso tra gli utenti di videogiochi, non tra i produttori...

Emilio Orione, Nizza Monferrato (AT)

Sei proprio convinto di quello che dici?

C'è anche lui

...Noi improvvisiamo i 2 poliziotti della serie CHIPS...

Gianni Manenti, Noto (SR)

Pubblico la tua «lettera» (cioè il tuo nome) per dovere di cronaca, ma i CHIPS lasciamoli stare eh... (non ti sarai mica offeso?).

Anche per questo mese abbiamo finito, ma non disperatevi... Ciao!

di Marco Pesce

Un videogioco tutto nostro

sesta puntata

Bene bene, anzi, male male. Evidentemente l'ho sparata grossa. Quel gioco non è affatto originale. Entriamo in meditazione
Cos'è che ci fa esclamare «... e bellissimo»? È un bel problemino questo.
Sinceramente non gli avevo mai dato eccessivo peso. A quanti di voi piace The Sentinel? Secondo me è un gioco fantastico. Mi permetto di qualificarlo in questo modo perché racchiude in sé due qualità fondamentali:
1) la grafica eccezionale
2) il concetto di gioco innovativo

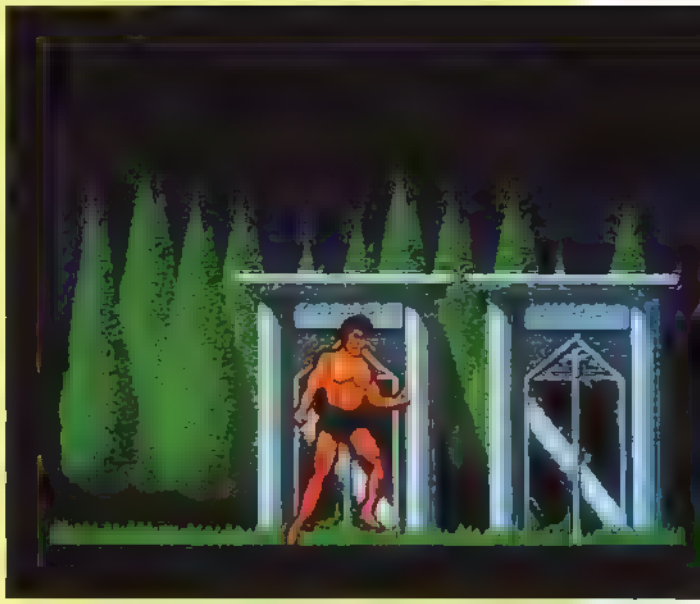
Qualcuno potrà obiettare che il punto 2) è verificato in conseguenza del punto 1), anzi, che il punto 2) è parte integrante del punto 1), mentre qualcuno invece non lo sa nemmeno cos'è The Sentinel, oppure non è assolutamente d'accordo con me. Eppure è un gioco che ha senz'altro richiesto uno sforzo enorme per la sua realizzazione, ma il non immediato impatto con il giocatore gli ha dato quell'aria di «ma cos'è 'sta roba». È questo che si merita l'originalità? Prendiamo in esame un classico gioco che viene giudicato come bel gioco dall'80% dei suoi giocatori. Questo ha un buona grafica, un sonoro accettabile, una struttura di gioco realizzata con lo stampino! Il computer è uno strumento molto potente e non pretendo di essere stato il primo a scoprirlo, ma dico io, possibile che con tutta la potenzialità che offre non riusciate a pensare a un concetto di gioco veramente nuovo? Inventare qualcosa di diverso dal solito non vuol dire scrivere una storia nuova e questo non lo avete per niente afferrato. Tutte, e dico tutte, le vostre proposte non sono altro che la revisione di qualcosa già vista. Prendiamo ad esempio il gioco di Paolo Costabel; è una simulazione ippica «... un tema poco sfruttato...» ma questo non lo annovera di certo tra le concezioni geniali. Così come il caro Paolo ce la presenta (vedi MC 71) è, anzi sarebbe, un bel gioco, ma non risponde alla mia richiesta di «... voglio un'idea nuova...» Non ho intenzione di prendermela con te, Paolo; le altre idee sono anche peggiori. Realizzare un Megagame non vuol dire solo far muovere 64 sprite contemporaneamente, tanto per dirla con una frase fatta. I computer giochi si sono fossilizzati. Un giocatore come me, che ormai ne ha viste di tutti i colori, non può non provare un senso di nausea osservando che tutte quelle che vengono spacciate per novità sono solo concetti di gioco stravecchi con una nuova storia. Sono sicurissimo che non avete afferrato quello che voglio dire. Non parlo del solito gioco spaziale (che ha stufato lo sanno anche i sassi, ma purtroppo non i programmatori come Martin Walker o Andrew Braybrook), ma del gioco che cerca di simulare la realtà, quindi anche giochi sportivi, di strategia militare, di avventura ecc. ecc.

I primi videogame basavano la loro forza sul meccanismo di gioco e questo perché non potevano certo impressionare con la bella grafica o con il bel sonoro (ci sono comunque diverse eccezioni). Poi le tecniche hardware si sono sofisticate; i miglioramenti hanno rinforzato esclusivamente il lato estetico. A peggiorare la situazione ci pensano i laser-game, ma i videogame-dipendenti si accorgono ben presto che c'è qualcosa che non va e la fine di queste macchine non tarda a manifestarsi. Il videogioco astratto è morto prima ancora di dimostrare la sua vitalità. Il videogioco non è più un argomento a se stante; ha bisogno di spunti presi dal cinema, dalla televisione, dalla vita comune, ma fate un errore se pensate che mi riferisco esclusivamente ai cosiddetti giochi su licenza. Siete autorizzati a urlare: «ma questo che cavolo cerca!»

Mi sono lasciato trasportare da meditazioni troppo profonde (!). Il punto è che giocare con un videogioco comincia a risultarmi noioso e sono alla ricerca di una scusa giustificatrice del mio stato di angoscia. Vi prego, se c'è qualcuno che la pensa come me o anche qualcuno che mi considera un imbecille è esplicitamente invogliato a scrivermi una lettera; un bel dibattito sull'argomento non dovrebbe guastare lo stomaco a nessuno.

Per chi fosse confuso, questa è la rubrica del Megagame 64. Quello che mi ha spinto a fare il lungo discorso introduttivo, oltre che le motivazioni già esposte, è stata la reazione dei lettori per quanto riguarda la mia «scelta di gioco». Sentite cosa dice Enrico Ferrante: «Non condivido il tuo entusiasmo per l'idea di Sergio di Fusco (MC 72 pag. 180), infatti pur presentandosi come Megagame poliedrico può anch'esso essere ricondotto ad un proge-

... E l'Amiga?



Le vedete queste tre schermate? Che ne dite? Per il «povero» Amiga di giochi così se ne vedono pochi, anzi purtroppo non se ne vedono affatto! Ma Marco Pesce non se ne sta certo con le mani in mano ad aspettare che qualcuno si decida a sfruttare questa potentissima macchina. Dalle vostre lettere ho scoperto che i possessori di questo computer sono molti di più di quelli che mi aspettavo. Mi hanno chiesto di «allargare» la rubrica e trattare anche di Amiga, ma mentre per il 64 ci sono tanti potenziali aiutanti, per la nuova Commodore la situazione è meno rosea. Ho studiato le sue possibilità e credo di essere in grado di realizzare un primo videogioco che la sfrutta in modo un tantino più adeguato di quanto è stato fatto finora. Questo videogame quindi sarà più che altro un esperimento, ma credetemi: cercherò a tutti i costi di farlo riuscire. Le tre schermate sono tratte per l'appunto da esso. Il lavoro procede tranquillamente e vi terrò informati sulle pagine di MC.

nitore che ne dici di Herbert's Dummy Run? Anche lì, nel contesto di un giochino divertente, si ritrovano tanti altri minigame. Certo l'originalità è difficile da raggiungere!»

Anche l'instancabile Paolo Costabel ■ dimostra contrario

«Ma come!? Hai tanto parlato di Megagame e poi liniamo a realizzare una trentina di minigiochi visti e rivisti? Sinceramente non ho nessuna voglia di giocare a trenta brutte copie di Space Invaders, Pac-Man etc. E poi come potrebbe un tale gioco rimanere nella storia del videogame? Gli mancherebbe il fascino e l'atmosfera di un gioco ben strutturato, con un tema ben preciso, un suo stile, un suo carattere. Inoltre se passasse questa idea ogni collaboratore si troverebbe necessariamente a scrivere un minigioco per i fatti suoi. Dove vanno a finire i bei discorsi sulla collaborazione e sulla solitudine del programmatore. E poi per i trenta giochini, saremmo di nuovo da capo. Se ci vogliono sei mesi per scegliere una proposta, figuriamoci quanto ci vorrà per trovare trenta idee diverse!».

Da queste due lettere traspare chiaramente che il nostro Megagame deve essere un gioco in grado soprattutto di dimostrare che anche l'Italia è «tecnicamente» valida nel campo della realizzazione dei videogiochi.

Prima di proseguire vorrei per lo meno tentare di giustificare la mia scelta. Un megagame-multigame può anche risultare un bel gioco, soprattutto se i giochi in esso contenuti sono tutt'altro che minigame. Trenta buoni giochi dinamici accontenterebbero il 90% dei giocatori e, come nelle mie intenzioni, se realizzati superbamente, risolverebbero anche il problema dell'«orgoglio italiano» (comincio a diventare monotono, lo so...) Visto che il concetto originale non è saltato fuori, meglio riversarsi su temi ben collaudati, pensavo io. Evidentemente la delusione, conseguenza del non aver trovato fra voi il genio dell'ideazione, deve avermi fatto cadere molto in basso. Mi rendo conto che tanti giochi di media qualità o, bene che vada, di discreta qualità non sono compatibili con le mie dichiarazioni. Meglio fare un gioco di qualità con sceneggiatura e struttura di gioco varie, anche se non eccessivamente originale. È arrivato il momento di smetterla con le titubanze. Cominciamo e basta. Lo liniamo e se poi non piace... cosa ci importa, possiamo sempre farne un altro, no?

Quale sarebbe questo gioco direte voi. Entrambi i lettori critici giudicano le idee di Franco Violante come «le più originali» e in particolare la prima. Enrico poi propone una sua idea praticamente identica ■ quella di Franco. Vediamo un po'. Il gioco consiste nel crearsi un impero galattico con l'ausilio della nostra abilità governativa. Siamo in una galassia costituita da un complesso di sistemi solari, comprendenti diversi pianeti più o meno abitati. Il nostro ambizioso progetto sarà contrastato da numerosi imprevisti e naturalmente anche da problemi di normale amministrazione. Il gioco ricalca lo stile di Defender of the Crown, ma, almeno nelle nostre intenzioni, dovrà essere un po' più articolato. Passiamo al dunque. Innanzi tutto dobbiamo studiare un sistema di comando per la gestione del nostro protagonista. Che ne dite di un grosso centro elettronico (tecnologie di casa nostra... è meglio) dove il giocatore (chiamiamolo... Paolo) praticamente vive? In questo luogo è possibile controllare l'intero pianeta centrale. Inoltre abbiamo a disposizione i centri di ricerca (armi, alimenti, energia, materiali, etc.), i centri di difesa/attacco, i centri di soccorso (per i problemi del popolo). Il controllo del pianeta si effettua grazie ad ampie strumentazioni e sofisticati circuiti di trasmissione dati. Per mezzo di queste attrezzature Paolo può comandare spedizioni spaziali di ricerca oppure difendersi da attacchi esterni, quindi organizzare strategie o anche soddisfare le richieste del pianeta centrale e di quelli colonizzati. Paolo si aggira tra i vari compartimenti e può liberamente osservare l'andamento del lavoro dei suoi dipendenti o prendere decisioni sul da farsi. Egli è munito di uno strumento portatile che lo avvisa in caso di allarme o in caso di richiesta di uno specifico settore. Grazie ad un veloce sistema di trasporto, può spostarsi senza difficoltà ■ raggiungere quindi la postazione desiderata. Chiaramente, nel caso la particolare situazione lo richieda, egli può partecipare direttamente all'azione come ad esempio nel caso di un'esplorazione ■ nel caso di uno «scontro a fuoco». Una bozza generale del gioco l'ho fatta. Possiamo passare a definire qualche dettaglio. Com'è facile intuire, non è possibile contenere il gioco tutto in memoria, se non riducendo le sue dimensioni. Occorrerà quindi suddividerlo in più file e con questo voglio dire che utilizzeremo anche il drive. Una versione su cassetta credo proprio sia impossibile da realizzare (... mi dispiace). D'altra parte comprimere il gioco in 64K si prospetta impresa tutt'altro che semplice, ma in ogni caso, se ci saranno lamentele, cercheremo di risolvere il problema. Dunque, per il mezzo di trasporto interno aveva pensato di realizzare una specie di tunnel con vista 3D. Impostata la destinazione su un quadrante, il nostro monoposto corre fino a destinazione. A questo punto la vista cambia e ci troviamo di fronte all'ingresso del settore scelto, la portiera scorre verso l'alto e si intravede l'interno. Possiamo entrare o proseguire il viaggio nel tunnel (anche Paolo sbaglia!). Se decidiamo di entrare la vista cambia ancora. Con scrolling orizzontale spostiamo il nostro eroe nel compartimento e possiamo avvicinarci agli strumenti di controllo, che danno informazioni sullo stato del settore o permettono di impostare comandi. Ci occorre ora una mappa dell'intero centro di controllo con descrizione delle possibilità di ogni settore:

1) settore centrale: permettere di osservare la mappa dell'intero centro e la mappa della galassia. È munito di un radar che avverte in caso di attacchi provenienti da pianeti esterni. È possibile comandare spedizioni spaziali sia di ricerca che di attacco o difesa.

2) Settore ricerche: in questo compartimento numerosi scienziati sfruttano le loro capacità nel tentativo di scoprire materiali utili alla vita del sistema. Armi più potenti, edifici più resistenti o anche nuove forme di alimenti o nuove medicine possono essere i risultati delle ricerche. Chiaramente, come accade per gli altri settori, sono necessari dei fondi economici. Possiamo quindi controllare lo stato delle ricerche e decidere se rinforzare questa attività. Se una particolare ricerca produce dei risultati verremo avvisati e quindi si potrà agire di conseguenza.

Ad esempio supponiamo che un determinato materiale venga ritenuto idoneo alla costruzione di nuove potentissime armi, ma che questo materiale si trovi solo in un lontano pianeta, bisogna decidere se è opportuno ■ meno effettuare una spedizione.

3) Settore soccorsi: questo centro è quello che si occupa dei problemi di normale amministrazione che non richiedono l'intervento del nostro Paolo, ma tuttavia necessitano anch'essi di risorse economiche.

4) Settore economico: ci mette di fronte alla gestione delle risorse del sistema e ci informa di eventuali soluzioni in caso di difficoltà.

5) Settore costruzioni.

6) Settore produzione cibo.

7) Settore produzione medicine.

8) Settore costruzioni strutture.

Dal settore 5) al settore 8) troviamo compartimenti dedicati ad attività specifiche che sono organizzate in maniera simile.

Hanno tutti delle richieste economiche e ci permettono di controllare la

situazione e di prendere decisioni in proposito.

Tutte queste attività vengono svolte sia per il pianeta madre che per quelli che man mano vengono colonizzati.

Bene, un piccolo passo avanti è stato fatto. Certo occorre ancora scendere nei dettagli veri e propri che poi ci permetteranno di realizzare il programma... vero e proprio. Sarà proprio questo il compito che ci aspetta nella prossima puntata ■ forse si potrà anche cominciare a parlare di programmazione (finalmente!).

Nel frattempo vi ricordo che se avete qualche consiglio o qualche aggiunta da propormi io sono sempre pronto ad ascoltarvi.

Scuola di videogame

Moltiplicazione degli sprite (seconda parte)

Dove eravamo rimasti?... ah! sì, «quindi alla prossima IRQ il microprocessore eseguirà il codice contenuto a partire dalla locazione $7030»

Dunque, come è possibile fare la stessa cosa in Basic? Basterebbe (notare il condizionale) «pokare» nella locazione $DC0D il valore #$7F e nelle locazioni $0314 e $0315 rispettivamente i valori #$30 e #$70. Purtroppo però non è possibile inserire queste due ultime poke senza provocare il blocco del computer, in quanto abbiamo trascurato la funzione dell'istruzione in LM "SEI" che come detto disabilita le IRQ e (come non detto) in Basic non esiste. Disabilitare le IRQ è necessario in quanto altrimenti anche in LM ci sarebbe il blocco del computer (non è detto, ma meglio premunirsi). Perché il C-64 si blocca? Supponiamo che, girando per Roma con la vostra automobile, non riuscite a trovare la strada giusta (non è poi così difficile). Un gentile signore si fa avanti ■ comincia a descrivervi la strada che dovete seguire. Vostro padre che è alla guida, però, non ha capito che state parlando con quel tizio e continua ad andare avanti. Non siete riusciti a capire bene l'informazione del signore ■ inevitabilmente vi perderete. La stessa cosa avviene nel Commodore. La macchina che continua ■ camminare sono le IRQ e voi che state ascoltando l'informazione siete il vettore di IRQ. Il microprocessore tenta di comunicarvi «la strada» ma non fa in tempo e l'interruzione salta a un vettore sbagliato! (spero di non aver complicato la comprensione di questo semplice concetto). Dicevamo che in Basic non esiste questa istruzione, ma non ■ un problema in quanto un programma del genere non viene mai scritto in Basic. Dunque SEI, LDA, STA, LDA, LDX, STA, STX,.. NOP! Cosa vuol dire NOP? Questa istruzione che occupa sempre solo un byte serve per riempire una zona di memoria quando quest'ultima è occupata da altre istruzioni inutili. Direte voi: «a cosa serve?». Serve quando quelle istruzioni sono di intralcio al programma e quindi devono essere ignorate. Sostituendole con delle NOP il risultato che si ottiene è che il microprocessore, incontrando le NOP, non fa niente e continua con la prossima istruzione. L'utilità di questo sistema si riscontra quando (come nel mio caso) un programma viene modificato e quindi ci si ritrova con codici superflui, come accade ad esempio se la modifica comporta la scrittura di un'istruzione che occupa due byte invece di tre. In questo caso il terzo byte sarebbe di intralcio al programma e magari comporterebbe confusione di istruzioni, quindi va sostituito con una NOP. Esempio pratico: supponiamo che (tenete d'occhio il listato del numero di aprile) a partire dalla locazione $7010 abbiamo scritto (invece della NOP) l'istruzione LDA $7840. Ci accorgiamo solo dopo aver già scritto il resto del programma che l'istruzione è errata. Apparentemente il rimedio è semplice; basterebbe sostituire a LDA $7840 l'istruzione giusta (nel nostro caso la LDA #$00). Quest'ultima istruzione però occupa 2 byte invece che tre come nel caso della LDA $7840. Se scriviamo «sopra» ad essa, l'ultimo byte resterà al suo posto e verrà interpretato come una normale istruzione (cosa che noi non vogliamo). Infatti il terzo byte (equivale al $78), corrisponde all'istruzione SEI che in tal caso non darebbe fastidio (perché due SEI di seguito non sono dannosi), ma poteva anche andarci peggio. Se ad esempio avevamo un $8D, equivalente al codice operativo dell'istruzione STA, i successivi due byte venivano interpretati come facenti parte della menzionata istruzione quindi veniva fuori l'istruzione STA $128D e il disastro era assicurato perché, oltre alla presenza dell'istruzione sbagliata, ci sarebbe stata anche la modifica delle successive istruzioni (meglio non parlarne). La soluzione migliore in questo caso è depositare nella locazione $7010 la NOP e scrivere a partire dalla locazione $7011 l'istruzione LDA #$00. Chiaramente era anche possibile traslare tutto il programma con un'istruzione da impartire al monitor-assemblatore, ma in questo caso è superfluo, soprattutto tenendo conto che non sempre è possibile eseguire questa operazione senza modificare alcuni punti del programma. Per il momento lasciamo stare e proseguiamo. A proposito, a partire dalla locazione $7025 fino alla 702F sono state inserite una serie di istruzioni NOP ma questo non vuol dire che avevo sbagliato tutto il programma. Le NOP sono utili anche nel caso in cui non sappiamo a priori se in seguito sarà necessario inserire altre istruzioni, quindi solitamente vengono utilizzate per «lasciare occupato il posto». Da tutto questo discorso traspare che a differenza di quanto avviene in Basic, l'inserzione di nuove linee di programma non è poi così semplice.

Torniamo al listato. Il resto del programma, contenuto dalla locazione $7011 alla locazione $7020, fa in modo che la prima IRQ si verifichi in corrispondenza della posizione «zero» del pennello elettronico e abilita le interruzioni del RASTER. In seguito riabilita le IRQ e ritorna (in questo caso) al Basic. Le locazioni $D012 e $D011 contengono la posizione del pennello elettronico, ma se usate in scrittura, permettono di impostare una particolare posizione del pennello elettronico che, quando verificata, genererà l'interruzione (che discorso contorto). In dettaglio, la locazione

$D012 contiene gli otto bit meno significativi e la $D011 contiene (tra le altre cose) il bit più significativo (in quanto 256 divisioni non sono sufficienti a ricoprire l'intera area del RASTER; 512 sono più che sufficienti). Per impostare la posizione «zero» occorre azzerare la locazione $D012 e impostare a zero il bit più significativo della locazione $D011. La prima operazione è semplice e la sappiamo fare tutti, mentre la seconda richiede l'introduzione di una nuova istruzione: la AND. Azzerare solo un bit infatti significa lasciare inalterati gli altri e il sistema migliore per fare ciò è quello di depositare il contenuto del registro nell'accumulatore, sottoporlo ad un'operazione di and logico e ridepositarlo nella sua locazione. Il valore #$7F in binario si scrive 01111111, cioè il bit più significativo è posto a zero. Supponiamo di avere un qualunque valore nel registro $D011, ad esempio 10 (cioè, in binario, 00001010). Un'operazione di and logico con il valore 01111111 produce come risultato il valore (binario) . 00001010. Praticamente il byte è restato inalterato, ma se invece di avere 00001010 avevamo 11111111, il byte diventava 01111111, cioè il bit più significativo veniva azzerato (come volevasi dimostrare). A titolo di esempio, se volevamo azzerare il bit meno significativo, bastava eseguire un and logico tra il valore 11111110 e quello contenuto nel registro $D011 (semplice no?). Non pretendo di farvi un corso di elettronica digitale, ma spero che la mia spiegazione sia stata ugualmente sufficiente. Torniamo all'istruzione AND. Essa è proprio quella che esegue l'operazione di and logico tra accumulatore e valore fornito. Basta un LDA $D011, AND #$7F, STA $D011 e l'azzeramento è effettuato. Le successive due istruzioni depositano il valore #$00 nella locazione D01A e cioè abilitano le interruzioni RASTER. Il CLI sappiamo già che serve a riabilitare le IRQ e infine l'RTS serve a tornare indietro (è simile al RETURN del Basic).

Dalla locazione $7030 in poi troviamo la famigerata routine IRQ (ora viene il bello). La funzione di queste istruzioni non dovrebbe essere tanto oscura, almeno per quello che riguarda il loro... funzionamento. È interessante invece osservare il risultato che esse producono. «Carica l'accumulatore con #$FF e mettilo in $D019». Questa operazione è la meno interessante, ma va ugualmente eseguita ad ogni IRQ. Seguono quattro istruzioni adibite al cambiamento della routine IRQ (di nuovo!). Sono molto simili a quelle usate la prima volta. È curioso osservare che questa volta le IRQ non vengono disabilitate, ma questo perché siamo già in IRQ (eh, eh, eh) e quindi non c'è pericolo. Le istruzioni seguenti sono ancora del tipo... «già viste». Servono a cambiare la posizione del IRQ raster che si verificherà ora alla posizione #$60. Da $704E in poi troviamo due istruzioni che azzerano o meglio pongono il valore 0 nella locazione $D020, il che (finalmente) produce qualcosa di visibile, cioè cambia il colore del bordo (e lo fa diventare nero). Locazione $7081, nuova istruzione: JSR. Velocemente: è l'equivalente dell'istruzione Basic GOSUB. In questo caso c'è un "GOSUB $70D0". Saltiamo anche noi a questa locazione e continuiamo la nostra spiegazione da lì, cioè continueremo la nostra spiegazione da lì visto che per questo mese direi che può bastare, no?..

Megaposta

Ci divertiamo eh? Sono arrivate ben otto lettere con la soluzione del giochino proposto dal famigerato Gianni Z. Sei di queste l'hanno risolto in maniera esatta, mentre le restanti due... hanno sbagliato! Mi sembra doveroso quindi elencare i nominativi dei lettori che sono riusciti nell'impresa. Permettetemi di cominciare con un gruppo di bambini che, con l'aiuto del loro maestro, hanno risolto il giochino a scuola. Sono gli alunni della terza classe delle scuole elementari «L. Ariosto» di Reggio Emilia:
Federica Melegari, Elisa Ferroni, Giuseppe Aiello, Salvatore Rivello, Ernesto Lettieri, Francesco Maseroli, Alessandro Spaggiari, Stefano Trotta

Cosa dire a questi cari bambini... BRAVISSIMI! Un bravo lo merita comunque anche il loro maestro, Raimondo Motti.

Proseguiamo l'elenco dei bravi:
Roberto Croci da Legnano (MI)
Paolo Costabel da Genova
Damiano Verzulli da Chieti Scalo (CH)
Riccardo Giannetti da Torrenieri (SI)
Marco Zuccarini Chieti Scalo (CH)

Ed ora che ne dite dell'elenco dei... meno bravi? Si tratta di Gianni Sarti, che dopo aver insultato tutta la redazione si pavoneggia con una soluzione sbagliata (ah, ah, ah). L'altro è Emanuele Aliberti, che per lo meno apre il discorso con un «credo di aver trovato la soluzione al problema...». La soluzione è apparsa sul numero precedente a quest'ora dovrebbero averla già letta...

Passiamo alle lettere.

Ma come hai fatto?

«... Anche il programmino più sciocco si traduce nel blocco totale del computer o nel migliore dei casi in un ritorno al Basic (ovviamente senza risultati). Vorrei chiedere perciò a Marco Pesce, che mi sembra molto tosto sull'argomento, come ha fatto a diventare così bravo, cioè quali libri ha letto e ritiene necessari e sufficienti per imparare le cose giuste e fare un po' di esperienza. Io dispongo di un Commodore 64 e di un Sinclar QL».

Roberto Croci, Legnano (MI)

Caro Roberto, com'è facile intuire che la mia esperienza (modestia a parte) non deriva solo da un profondo studio teorico. Essa è il frutto di molta pratica e di molti tentativi andati a male. Nel campo informatico è necessario fare così e tu dovresti saperlo. Non è il caso quindi di allarmarsi se il nostro programmino LM non gira al primo colpo. A proposito, è opportuno munirsi di tasto RESET, il blocco di un programma in linguaggio macchina non perdona. Cosa consigliarti quindi se non di seguire le mie «lezioni» e i miei suggerimenti?

Voglio uno scrolling!

«Sono consapevole dell'insufficienza di spazio da dedicare alla scuola di Videogame, ma non mi spiacerebbe un microscopico listatino in LM sullo scrolling in 2D...».

Roberto Ricci, Torino

Accontenteremo Roberto? Ma certo e accontenteremo anche qualcun altro un po' più esigente.

Offresi programmatore

«Idee per un videogame onestamente non ne ho, anche perché penso che ormai tirar fuori qualcosa di nuovo sia molto, ma molto arduo. Posso comunque offrire, se necessario, la mia esperienza di programmatore sia Basic che LM (v. per esempio AlfaDisk e NL-Printgraf pubblicati su MC, più molti altri articoli su CCC e Personal Computer)...».

Roberto Morassi, Pistoia

Preparati Roberto...

Per finire ringrazio inoltre:
Marco Paolini di Cernusco sul Naviglio (MI), Giuliano Peritore di Latina, Andrea Beltrame di Novi Ligure (AL), Nicola Marangon di Cappella di Scorze' (VE), Luca Sassone di Caronno P.lla (VA), per aver inviato il loro contributo.

Tanti saluti a tutti.

di Marco Pesce

Un videogioco tutto nostro

settima puntata

Ragazzi, avete mandato le vostre lettere? Speriamo di si altrimenti questo Megagame diventa un Marcogame. Per questa puntata ho fatto tutto da solo, ma sono in attesa di vostri suggerimenti. Dov'è finito il vostro entusiasmo per l'iniziativa?

Cosa avevo promesso di fare... ah si; «...occorre scendere nei dettagli veri e propri...». Come potete osservare dalle foto di questa pagina, ho abbozzato un po' di grafica, tanto per avere un'idea di come saranno le scene. Intendiamoci, non ritengo assolutamente opportuno considerarle come le migliori schermate mai viste sul 64. Tra l'altro le ho realizzate in fretta e furia e non pretendevo di generare dei capolavori. Un grazie va al mio Amiga 500 e al Deluxe Paint, che quando si tratta di fare le cose al volo è di grande aiuto. Non preoccupatevi, nel disegnare ho tenuto conto di quelle che sono le limitazioni del C-64... almeno in gran parte.

Torniamo al gioco. Prima di iniziare a scrivere questo articolo ho buttato giù parecchi schizzi e ho effettuato diverse modifiche. Lo scopo del gioco deve essere definito con più chiarezza. La galassia nella quale è situato il nostro pianeta comprende 20 sistemi solari (!), ognuno dei quali è costituito da un numero variabile di pianeti che va da un minimo di 4 a un massimo di 16. Lo scopo del gioco è quello di conquistare tutti i pianeti abitabili (chissà cosa ce ne faremo poi...). Cominciamo con il descrivere come dovrà essere il settore centrale. Tramite un computer e una mappa abbiamo il contatto con i pianeti della galassia. Quando ho disegnato la foto numero 1 pensavo appunto a come poteva essere il settore centrale. La freccina gialla fa da puntatore. Se il puntatore viene posizionato (tramite joystick) sul pannello con le due frecce, la mappa scorre orizzontalmente ed è possibile quindi osservarla per intero. Spostando la freccina su una delle stelle (che nella schermata sono rappresentate dai pianeti più grandi) il computer entra in funzione e ci comunica il nome della stella e il numero dei pianeti del sistema solare che essa costituisce. Non tutti i pianeti sono abitabili in quanto non tutti possono giovarsi dei benefici prodotti dal loro sole, tuttavia qualunque pianeta può possedere delle risorse energetiche, magari sconosciute. Ogni pianeta quindi possiede le sue caratteristiche che possiamo riassumere brevemente qui sotto:

1 - tipo di pianeta (abitabile o non abitabile)
2 - quantità di energia posseduta
3 - possesso o meno della base per la produzione di energia
4 - abitato o non abitato
5 - numero di abitanti
6 - tipo di popolazione (nemici o amici)
7 - rapporto di risorse/abitanti
8 - rapporto spazio/abitanti
9 - quantità di fondi economici
10 - quantità di forze armate.

È chiaro che essendo considerevole il numero di pianeti, occorre risparmiare più memoria possibile. I dati 1, 3, 4, 6 possono agevolmente essere contenuti in un solo bit ciascuno; considerando un byte intero abbiamo ancora a disposizione 4 bit. I dati 2, 7, 8, 9, 10 li facciamo entrare in 4 bit ciascuno; fanno 2 byte e mezzo, ma il mezzo byte in più lo mettiamo nel byte che ci era avanzato prima. Resta il dato 5 che andiamo ad inserire in un byte intero; totale: 4 byte. Ricapitolando, 4 byte per pianeta moltiplicati per una media di 10 pianeti a stella moltiplicati per 20 stelle fanno... 800 byte. È da notare che è possibile risparmiare un'ulteriore quantità di memoria considerando che per i pianeti non abitabili sono sufficienti solo i dati dall'1 al 4, ma per ora lasciamo stare. A cosa servono questi dati? Procediamo con ordine. Ogni pianeta deve essere gestito in parte dal pianeta centrale. Tramite il computer del settore centrale noi possiamo:

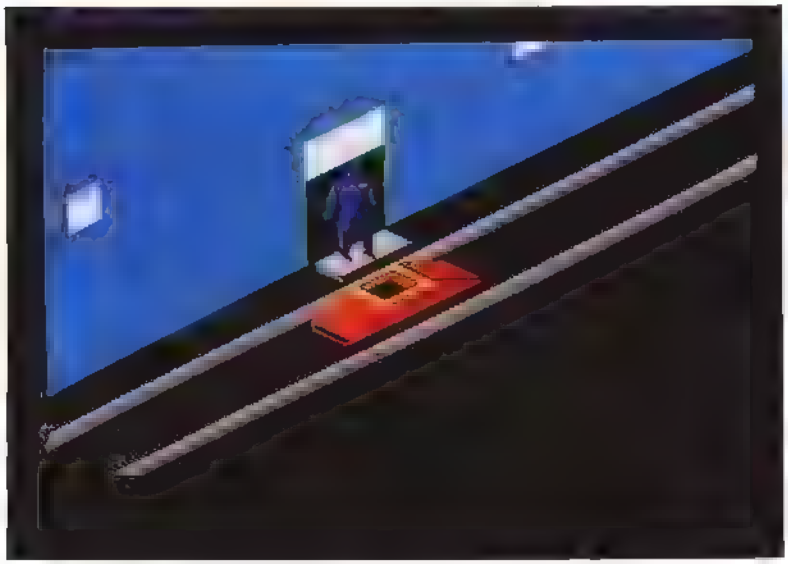
1 - trasferire la popolazione da pianeta a pianeta
2 - trasferire le forze armate da pianeta a pianeta

Foto 1

3 - trasferire energia o fondi economici da pianeta a pianeta
4 - ordinare la conquista di un nuovo pianeta
5 - contribuire alla difesa di un pianeta attaccato
6 - ordinare una spedizione di ricerca.

Tutti i dati servono per l'appunto a quantificare queste operazioni. È lecito ora chiedersi per quale motivo è necessario effettuare queste operazioni. Il punto 4 e il punto 5 sono quelli indubbiamente più immediati, in fondo è questo lo scopo del gioco. Gli altri punti sono necessari proprio per sostenere queste due attività principali. Il punto 1 è fondamentale per ottenere la massima produzione da un pianeta, in quanto essa si raggiunge solo quando c'è un giusto equilibrio spazio/abitanti. Il punto 2 è utile nel caso in cui un determinato pianeta corre più rischi di essere attaccato rispetto ad un altro e quindi necessiti a maggior ragione di forze armate. Per il punto 3 è necessario fare una piccola discussione. Ogni pianeta produce (ovviamente se è abitato) una certa quantità di fondi economici più una certa quantità di energia; nel caso venga conquistato un nuovo pianeta è possibile che quest'ultimo non sia in possesso dei requisiti necessari per essere considerato autonomo (almeno per quanto riguarda la produzione menzionata). È necessario in tal caso trasferire parte dei possedimenti di un pianeta più ricco per soddisfare il fabbisogno del pianeta conquistato. A questo punto bisogna ricordare che un trasferimento, qualunque esso sia, richiede un certo impiego di capitali quindi non è consigliabile effettuarlo ad occhi chiusi. Consideriamo infine il punto 6. La spedizione di ricerca si effettua quando uno dei centri di ricerca scopre una nuova forma di energia o un nuovo materiale utile per un determinato scopo. In tal caso la spedizione sarà necessaria al fine di costruire una base sul pianeta o sui pianeti ritenuti ricchi di tali elementi.

Foto 2



A proposito dei centri di ricerca, la volta scorsa li avevo racchiusi sotto un unico compartimento mentre è opportuno suddividerli sia perché effettuano attività diverse, sia perché altrimenti i settori del centro di controllo sono troppo pochi (e il tunnel è sprecato!).

Avremo quindi le ricerche di:
— medicine
— cibo
— energia
— armamenti
— costruzioni

ognuna delle quali potrà essere incentivata o meno da un sovvenzionamento economico. La ricerca di medicina è necessaria in quanto la popolazione deve essere la più numerosa possibile e la scoperta di una medicina più potente sarebbe un ottimo sistema per incrementarla. La ricerca di cibo invece va intesa nel senso che il cibo costa e riuscire a trovare una forma più economica non può che giovare all'economia dell'impero. La ricerca di nuove forme di energia si commenta da sola. Per quanto riguarda gli armamenti questa attività può scoprire il sistema per rendere più potenti le nostre astronavi. Infine le costruzioni, che se più resistenti daranno del filo da torcere agli attacchi delle popolazioni nemiche.

Ritorniamo al settore centrale. Per avere la situazione sotto mano sarebbe opportuno che il computer ci venisse incontro quando dobbiamo eseguire delle scelte, quindi, visto che in esso sono contenuti tutti i dati necessari, è possibile realizzare un sistema di analisi rapida del tipo «quanti sono i pianeti che...» tafferato il concetto!. Detto computer si trasforma quindi in un Data base vero e proprio al quale è possibile chiedere:
— quanti sono i pianeti in nostro possesso
— l'elenco dei pianeti in nostro possesso
— quanti e quali sono i pianeti popolati in maniera ottimale
— quanti e quali sono quelli popolati insufficientemente
— quanti e quali sono quelli popolati in abbondanza
— quanti e quali sono quelli riforniti in abbondanza di energia
— quanti e quali sono quelli riforniti insufficientemente
— quanti e quali sono quelli riforniti in modo ottimale

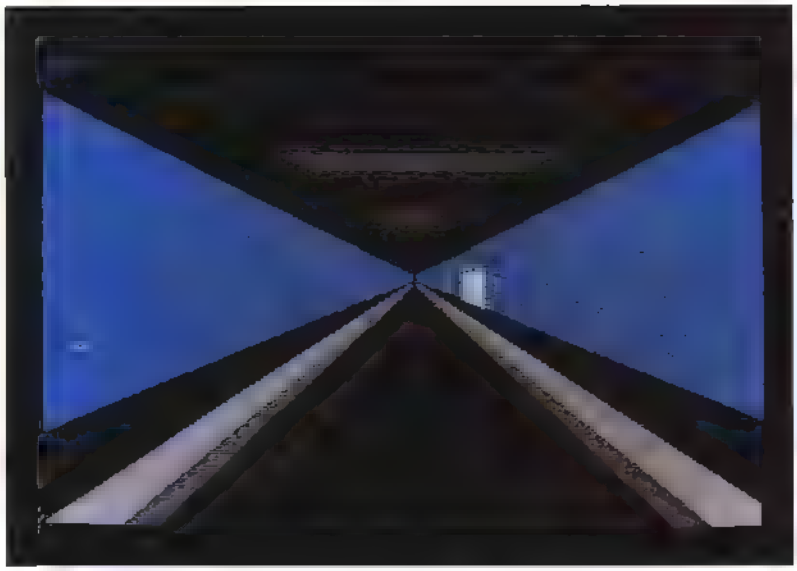


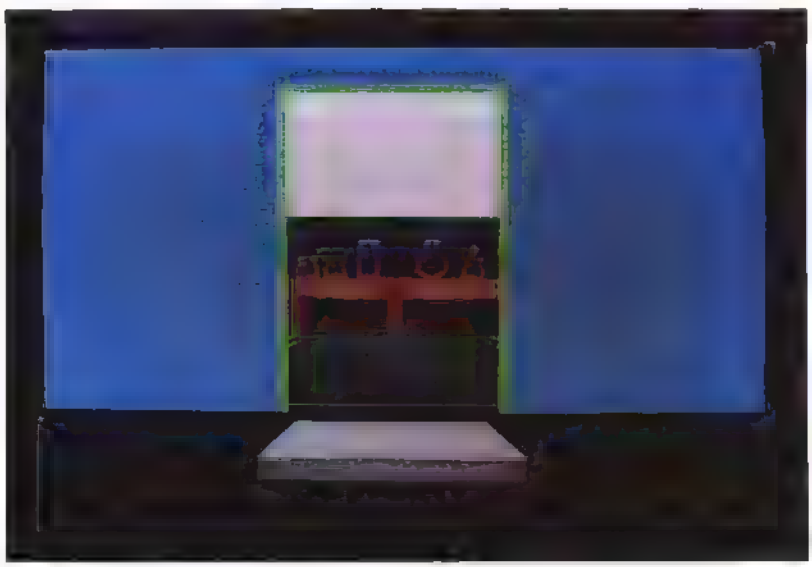
Foto 3

Ce ne andiamo a spasso per il settore (niente foto...) e poi usciamo. Paolo rientra nel suo «macinino» e via.. Siamo giunti così anche questo mese alle conclusioni anche se la mia promessa non è stata mantenuta per intero. Di tempo ne abbiamo ma non sprechiamolo troppo e mi rivolgo soprattutto a voi. Se non siete rapidi le vostre lettere rischiano di arrivare in ritardo e così va veramente a finire che il gioco lo faccio solo io. Per quello che riguarda la grafica vi concedo ancora un po' di tempo. Per quello che riguarda la struttura di gioco cercherò di completarla entro i prossimi due mesi. A buon intenditor poche parole.

e quindi impostare i giusti comandi senza dover per forza scandagliare tutta la mappa.

Osserviamo le altre tre foto. Cosa rappresentano? Ho effettuato una piccola modifica alla scena nel tunnel. La visuale è inizialmente quella della foto 2, quindi una prospettiva isometrica. Paolo entra nel mezzo di trasporto (si è proprio quello) e decide quale direzione prendere. Il menzionato monoposto comincia la sua corsa. Il fondale è fisso mentre accelera. Raggiunta una certa velocità la vista cambia e si sfreccia nel tunnel con vista in prima persona (foto 3). Giunti nei pressi della destinazione si ritorna alla scena della foto 2 e il monoposto rallenta fino alla porta. Una volta fermatosi la scena cambia con quella di foto 4. La porta comincia a sollevarsi e osserviamo l'interno del compartimento. Entriamo? e la vista cambia ancora!

Foto 4



Scuola di videogame

Moltiplicazione degli sprite (3)

Finalmente vedo il listato sulle pagine di MC. E già perché con quel fogliaccio lungo che mi ritrovavo era facile sbagliarsi e saltare qualche istruzione... è successo!!! Sfruttando la confusione generale che si era generata sul mio tavolo il diavoletto è riuscito a metterci lo zampino e mi ha fatto confondere. Dov'è l'errore direte voi (... se non siete dei lettori attenti); le ultime parole famose... «... Locazione $7081, nuova istruzione: JSR...». La locazione $7081 non c'entra proprio niente; la nuova istruzione infatti si ritrova già a partire dalla locazione $7051 ed è proprio a questa locazione che dovevano andare i miei occhi un mese fa. Chiarito l'equivoco (almeno spero) possiamo riprendere il nostro discorso. Dunque, alla locazione $7051 troviamo quindi la prima JSR che in questo caso salta alla locazione da $70C0. La routine memorizzata nelle locazioni da $ 70C0 a $70CB è molto simile alle altre due routine che seguono (in ordine di locazioni), memorizzate da $70D0 a $70DB e da $70E0 a $70EB e infatti servono per eseguire su per giù lo stesso compito (... su per giù). Esaminiamo la prima. Locazione $70C0 LDX # $10; nessun problema ma ricordiamoci di quanto abbiamo fatto (mettiamo # $10 in X). L'istruzione successiva è già più impegnativa; cosa vuol dire quel ",X" dopo il ben noto LDA $7100. Cosa abbiamo messo in X? (l'ho detto 2 secondi fa). Se vi dicessi che nell'accumulatore, invece della locazione $7100, viene caricata la locazione $7110 mi credereste? cretedemi. Questa «nuova» istruzione quindi somma il valore del registro X al valore fornito dall'istruzione. Volendo fare la stessa cosa in Basic bisognerebbe scrivere:

A= PEEK ((locazione)+X)

dove (locazione) è un valore qualsiasi mentre X è una semplice variabile. Ricapitolando il registro X viene usato come «puntatore». Continuiamo la lettura del

programma. La prossima istruzione è simile a quella appena vista ma è applicata alla funzione STA, quindi in Basic sarebbe:

POKE [locazione] + X, A

Andiamo avanti anche se non abbiamo ancora afferrato a cosa serve questa benedetta routine. Nuova istruzione fresca fresca: DEX. Guarda caso questa istruzione decrementa il contenuto del registro X. La solita analogia con il Basic ci fa scrive:

X=X-1

A questo punto troviamo un'istruzione veramente strana: BPL $70C2. Proprio qui sta il succo di tutta la routine, è tratta di un salto condizionato (!?). Niente paura, mi appresto a spiegarvi cosa significa. Come dice la parola stessa è quindi come è intuibile, un salto condizionato è un salto che per essere effettuato ha bisogno che una determinata condizione sia verificata. In questo caso il salto, al verificarsi della condizione, avverrebbe alla locazione $70C2 quindi il computer ricomincerebbe ad eseguire le istruzioni già viste e se non fosse del tipo «condizionato» non «uscirebbe» mai dalla routine. La condizione in questo caso è: «salta se il risultato della precedente istruzione ha prodotto un numero maggiore di zero». Quindi osserviamo il risultato prodotto dall'istruzione precedente: è sicuramente maggiore di zero, visto che è stato decrementato il registro X che prima era a $10 (ora sarà $0F quindi è ben lontano dall'essere «minore di zero»). Il salto quindi in questo caso viene effettuato e le istruzioni viste prima vengono rieseguite fin quando X viene decrementato dal valore zero al valore inferiore (che per ora non diciamo qual è). In tal caso l'istruzione BPL non si farebbe scrupoli e non darebbe luogo al salto. Ci sarebbe quindi il ritorno al programma «principale» grazie all'RTS. Il gruppetto di istruzioni quindi viene eseguito per ben 16 volte prima di «cambiare aria». A questo punto fate intervenire la logica.

Il programmino viene eseguito ogni volta con un valore di X diverso, quindi il valore che si addiziona alle istruzioni LDA $7100, X e STA $D000, X cambia di volta in volta. Questo vuol dire che i valori contenuti dalla locazione $7100 alla $7110 vengono trasferiti alle locazioni dalla $D000 alla $D010, ma forse ci eravate arrivati già da soli. Vogliamo capire a cosa serve tutto questo? Dalla locazione $D000 alla $D010 troviamo i registri delle coordinate degli otto sprite che in questo caso verrebbero alterate. Ricordate quando dicevo che questa routine è ripetuta in memoria tre volte ma con una piccola modifica? Se andiamo ad osservare le due routine seguenti ci accorgiamo che l'unica cosa che cambia tra loro, oltre la posizione in memoria e quindi i salti delle BPL, è il valore che segue la LDA, che nei tre casi è, rispettivamente, $7100, $7120 e $7140. Questi tre valori delle routine corrispondono quindi a tre cambiamenti di coordinate, che se effettuati nella stessa pennellata video danno origine alla moltiplicazione degli sprite! (8x3 = 24). Ovviamente questi tre cambiamenti devono essere effettuati al momento opportuno e quindi le tre routine sono gestite dalle tre rispettive routine IRQ che fanno in modo che il cambiamento si verifichi ad una ben determinata posizione del pennello elettronico. Se infatti da bravi bambini vi andate a guardare il resto del programma che non abbiamo ancora esaminato capirete come si svolge il trucchetto. Molti di voi avranno finalmente afferrato come funziona questo benedetto moltiplicatore di sprite e si meritano un «bravo». Agli altri devo dare appuntamento alla prossima puntata perché per questo mese non c'è proprio più tempo. Vi ricordo comunque che la prossima volta cominceremo anche un nuovo argomento. Saluti a tutti.

Megaposta

Gentilissimo Marco, dopo aver letto la Megaposta sul numero 73/aprile, ho deciso di inviare due righe per rispondere in maniera adeguata ad Alessandro Merolli di Roma.

Carissimo Alessandro, qui nessuno si crede di essere un EINSTEIN!!! Il fatto che tu non riesca a capire e/o risolvere un gioco matematico non implica che tutti siano degli incapaci, ma è soltanto indice della tua insufficiente capacità d'analisi dei problemi.

Come hai potuto constatare la soluzione era alquanto semplice! Mi scuso per aver abusato di questa rubrica per una comunicazione personale, ma era l'unico modo possibile per contattare Alessandro.

Gianni Zambon, Treviso

Aspettavo la tua replica Gianni... ma non sognatevi di continuare la discussione su queste pagine...

Per questo mese la Megaposta è stata tutt'altro che mega, ma purtroppo non ho il tempo per rispondere alle altre lettere e l'unica che mi sembrava «degna di urgenza» l'ho pubblicata. Scusatemi.

Il più grande e potente foglio elettronico per personal computer della nuova generazione, è anche il più veloce, il più facile, il più bello.

Quando vedrai di cosa è capace Microsoft Excel ti chiederai come hai potuto farne a meno finora.
Microsoft Excel non è solo il miglior foglio elettronico esistente: è meraviglioso. Adesso, parla italiano in due versioni; per personal computer MS-DOS e per Apple Macintosh. Sempre più meraviglioso.

Il miglio strumento analitico
Con Microsoft Excel le dimensioni del foglio non hanno più praticamente confini: ben 256 colonne per 16384 righe per sfruttare tutta la potenza dei personal computer della nuova generazione, creare modelli molto grandi e offrire pieno supporto alle espansioni di memoria: poi Microsoft Excel ti impressionerà per la sua velocità e per la sua intelligenza nel ricalcolare il foglio solo dove e quando serve e per la possibilità di richiamare sullo schermo più fogli contemporaneamente e stabilire collegamenti tra loro. E se credi che sia solo un foglio elettronico ti sbagli: sono state integrate anche potenti funzioni di database.

Il miglior strumento per la presentazione
Scegli adesso uno dei tanti

di Marco Pesce

Un videogioco tutto nostro

Particolarmente ridotta è la puntata di questo mese. Il motivo è presto detto: sto aspettando i vostri suggerimenti. Nel frattempo mi gingillerò con la esposizione di alcune probabili aggiunte per il Megagame

Ogni pianeta centrale avversario ha un centro di controllo simile al nostro. Distrutto detto centro l'impero relativo a quel pianeta perderebbe il comando. Una volta individuato il pianeta centrale di un avversario il giocatore può inviare una serie di robot molto potenti che dovranno addentrarsi in tale pianeta. Uno di questi robot sarà comandato direttamente dal giocatore. Avremo quindi un gioco sullo stile Ghosts'n goblins o Rastan che sarà lo sfogo di quelli che, come me, non disdegnano una partita a questo genere di giochi, specie se ben realizzato. E se la missione fallisce... fate un po' voi.

Ghosts'n Goblins

Rastan

Come potrebbe terminare una partita al Megagame?
1) Restiamo senza pianeti;
2) ci finiscono tutte le risorse;
3) ci distruggono il pianeta centrale

Per quello che riguarda gli spostamenti, stavo pensando che il loro costo poteva essere influenzato da:
1) la quantità di materiale da trasportare
2) il tempo necessario per il completamento del viaggio.

A proposito del tempo: le spedizioni devono tener conto anche di questo fattore, in quanto i «rifornimenti» potrebbero anche arrivare in ritardo! Quindi nel caso di una guerra in un pianeta lontano è opportuno pensarci due volte prima di inviare dei rinforzi (sempre che siano necessari), perché potrebbe anche accadere che nel frattempo il conflitto sia già terminato. Chiaramente i tempi di «volo» non saranno reali, ma (siamo in un gioco) avverranno a velocità molto, ma molto, più grande di quella della luce. Questa del controllo del tempo in definitiva è una funzione da aggiungere al grande data-base del nostro centro. Un'altra aggiunta da fare è quella relativa al controllo della quantità di armi distribuita in ciascun pianeta in nostro possesso (che lo scorso mese mi era sfuggita). Questo è tutto quanto sono riuscito ad aggiungere senza... aggiungere troppo.

Nell'attesa dei vostri imminenti «soccorsi» lascio la parola alla Scuola di videogame.

Scuola di videogame
... Moltiplicando sprite

Questo mese concludiamo il discorso portato avanti nelle ultime tre puntate con l'esposizione di altre due tecniche relative alla moltiplicazione degli sprite

Se infatti il listato proposto precedentemente è in grado di effettuare questa delicata operazione è anche vero che esso presenta notevolissime limitazioni. Il principale inconveniente è quello che i 24 sprite non possono essere posizionati liberamente nell'intera area dello schermo. Il motivo di questa mancanza è molto semplice ed è dovuto alla particolare tecnica utilizzata. Il programma effettua una divisione dello schermo in tre «zone», in ognuna delle quali un gruppo di otto sprite è libero di «vagare». Un'operazione del genere è necessaria in quanto, come noto, il 64 non può visualizzare più di otto sprite se non riutilizzandoli nella stessa «pennellata video», in posizioni diverse (cfr. MC n. 71; pag. 176). Il programma quindi è stato studiato per sfruttare questa caratteristica, ma lo fa nel modo più elementare possibile e cioè impostando tre diverse interruzioni raster ed effettuando, al verificarsi di ciascuna di esse, un cambiamento delle sole coordinate degli otto sprite, che devono necessariamente essere limitate in modo da far visualizzare ogni gruppo di sprite nell'area ad esso dedicata dalle due IRQ tra le quali è compreso. Si ha quindi la visualizzazione di 24 sprite, ma questi non hanno né forma né colori indipendenti da gruppo a gruppo e quindi risultano gemelli ■ triplette. Tuttavia quest'ultimo «inconveniente» non dipende dalla tecnica utilizzata, ma solo dalla semplificazione fatta al listato per renderlo comprensibile anche ai meno addentrati nel linguaggio macchina del C-64. Tenete presente infatti che per molti lettori è stata la prima «lezione» e a mio avviso siamo partiti anche troppo di corsa.

Prima di menzionare le tecniche alternative vorrei concludere il discorso della puntata precedente.

Ricordate quando parlavamo della BPL? Dicevamo che si trattava di una istruzione che eseguiva un «salto condizionato». La figura 1 mostra un elenco completo delle istruzioni del 6510 con relativo breve commento della loro funzione. In questo elenco troviamo altre sette istruzioni simili alla BPL, ma aventi ognuna un «tipo di condizionamento» diverso. Queste istruzioni effettuano le loro scelte basandosi sui valori contenuti nei bit del registro di stato (status register). Nel numero di aprile avevo promesso che avrei riparlato di questo particolare registro... Si tratta per l'appunto di un «byte» che contiene dei «bit segnalatori», detti anche «flag». La figura 2 illustra la posizione di ognuno di questi flag. Essi vengono alterati ogni qualvolta viene eseguita una istruzione che ne implica l'intervento. Ad esempio la «LDA» altera i contenuti del flag N ■ del flag Z. Il flag N è il segno del byte caricato nell'accumulatore che corrisponde al valore del bit più significativo (l'ottavo). Il flag Z indica (quando è posto a «uno») che il byte caricato ha contenuto nullo (uguale ■ «zero»).

A proposito del flag di segno (N): per rappresentare numeri con segno negativo il 6510 (ma anche molti altri microprocessori) utilizza la tecnica del complemento a due, che consente di rappresentare, con un solo byte, numeri che vanno da −128 ■ +127. Detta

Registro di stato (status register)

7	6	5	4	3	2	1	0
N	V	-	B	D	I	Z	C

7) Segno
6) Overflow
5) NON UTILIZZATO
4) Break
3) Decimale
2) Interrupt
1) Zero
0) Carry

Figura 2

Esempi di numeri negativi ottenuti con la tecnica del complemento a due

Numero positivo		Inversione dei bit	Numero negativo
Decimale	Binario		Addizione di "uno"
10	00001010	11110101	11110110
2	00000010	11111101	11111110
1	00000001	11111110	11111111

Figura 3

tecnica attribuisce convenzionalmente al bit più significativo di ogni byte il «segno» del valore che esso rappresenta. È chiaro quindi che così facendo il valore numerico è espresso dai soli primi 7 bit. Con sette bit sono possibili 128 combinazioni quindi se il bit più significativo è a zero il valore rappresentato può variare da 0 a 127, viceversa se il bit più significativo è uno il valore rappresentato varia da −128 a −1. È da notare che non è sufficiente cambiare il valore del bit più significativo per invertire il segno del numero in questione (ad esempio per passare da +5 a −5). Infatti per eseguire questa operazione con la tecnica del complemento a due occorre invertire tutti i bit del byte e aggiungere «uno». La figura 3 mostra alcuni esempi di inversioni di segno. Un sistema pratico per effettuare questo calcolo consiste nel sottrarre da 255 il valore del quale si vuole ottenere il complemento a due e poi aggiungere 1. Qualcuno potrà anche chiedersi il perché di tutta questa macchinosità e soprattutto l'utilità della cosa. Vi ricorderete senz'altro la subroutine presa in esame la puntata scorsa. Essa eseguiva un LOOP (ciclo) utilizzando il registro X come contatore. Quando il registro X diventava «più piccolo di zero» il loop finiva. Il valore immediatamente più piccolo di 0 è −1, che in complemento a due si scrive 255. Infatti decrementando di uno un byte avente valore uguale a zero si passa al valore 255 (come volevasi dimostrare!). Dopo questo esempio pratico dovrebbe essere chiaro che il complemento a due non è solo un sistema utilizzato per confonderci le idee.

Torniamo alle istruzioni di salto condizionato al registro di stato. Come detto questo registro memorizza degli indicatori; ora vedremo che cosa «indicano». Il flag N e il flag Z sono già stati esaminati. Il flag V (overflow) indica che c'è stato un «riporto» dal bit 6 al bit 7. è utile quando si effettuano delle operazioni in complemento a due (addizioni o sottrazioni) per controllare un eventuale errore nel risultato prodotto. Infatti nel caso in cui detto bit vale «1» ciò signifi-

Figura 1

Elenco delle istruzioni del 6510

ADC: somma il contenuto di un indirizzo di memoria con l'accumulatore, più il flag di carry.
AND: effettua un AND logico tra l'accumulatore e il dato fornito.
ASL: sposta tutti i bit contenuti nell'accumulatore, o in una locazione di memoria, di un bit verso sinistra. L'ottavo bit viene depositato nel flag di carry mentre il primo viene sostituito con uno «zero».
BCC: effettua un test del flag di carry. Se il flag è a «zero» effettua una «diramazione». Se è a «uno» continua il programma con l'istruzione successiva.
BCS: simile alla «BCC», ma opera in modo opposto, ovvero dirama se a «uno» e continua se a «zero».
BEQ: effettua un test del flag zero. Se è a «uno» dirama, altrimenti prosegue.
BIT: esegue un AND logico tra l'accumulatore e il dato fornito, ma non memorizza il risultato. Questa operazione tuttavia altera il flag di zero (lo pone a «uno» per un AND che dà risultato zero, viceversa lo pone a «zero») i flag «Overflow» «segno» nei quali deposita, rispettivamente, il bit 6 il bit 7 del risultato.
BMI: effettua una diramazione se il flag di segno è settato.
BNE: esegue una diramazione se il flag di zero non è settato (operazione inversa di «BEQ»).
BPL: esegue una diramazione se il flag di segno non è settato (operazione inversa di «BMI»).
BRK: genera un interrupt forzata. Il contenuto delle locazioni $FFF e $FFFF viene depositato nel PROGRAM COUNTER, mentre il contenuto di quest'ultimo viene depositato nell'area di stak. Anche il registro di stato è memorizzato nell'area di stack, con il flag di BREAK settato.
BVC: controlla il flag di overflow. Se non è settato effettua una diramazione.
BVS: simile alla «BVC», ma dirama se il flag di overflow è settato.
CLC: azzera il flag di carry.
CLD: azzera il flag D.
CLI: azzera il flag I.
CLV: azzera il flag di overflow.
CMP: sottrae il dato fornito al contenuto dell'accumulatore, ma senza depositare il risultato. Tuttavia i flag «SEGNO», «ZERO» «CARRY» vengono normalmente alterati.
CPX: come la «CMP», ma la sottrazione avviene tra il dato fornito e il registro X.
CPY: come la «CPX», ma la sottrazione avviene tra il dato fornito e il registro Y.
DEC: il valore contenuto nella locazione di memoria specificata viene decrementato di «uno».
DEX: simile alla «DEC», ma decrementa di «uno» il registro X.
DEY: simile alla «DEX», ma decrementa il registro Y.
EOR: effettua un «OR-ESCLUSIVO» tra il valore fornito e l'accumulatore.
INC: simile a «DEC», ma esegue l'operazione inversa (incremental).
INX: simile a «INC», ma opera sul registro X.
INY: simile a «INX», ma opera sul registro Y.
JMP: effettua un salto a una nuova locazione di memoria.
JSR: effettua un salto a una nuova locazione di memoria, ma, a differenza della «JMP», ritorna «indietro» quando incontra una «RTS».
LDA: deposita il dato fornito nell'accumulatore.
LDX: simile alla «LDA», ma deposita nel registro X.
LDY: simile alla «LDX», ma deposita nel registro Y.
LSR: simile alla «ASL», ma lo spostamento avviene verso destra e nel flag di carry viene depositato il valore del primo bit. L'ottavo bit viene azzerato.
NOP: non effettua alcuna operazione.
ORA: esegue un OR logico tra l'accumulatore e il dato fornito.a
PHA: deposita nell'area di stack il valore contenuto nell'accumulatore senza alterare quest'ultimo.
PHP: simile alla «PHA», ma deposita il registro di stato.
PLA: simile alla «PHA», ma riprende un valore dall'area di stack lo deposita nell'accumulatore.
PLP: simile alla «PLA», ma il valore viene depositato nel registro di stato.
ROL: simile alla «ASL», ma nel primo bit viene depositato il valore contenuto nel flag di carry prima che in quest'ultimo venga depositato il valore dell'ottavo bit.
ROR: simile alla «ROL», ma lo spostamento è effettuato verso destra.
RTI: il registro di stato e il program counter vengono alterati con valori provenienti dall'area di stack.
RTS: utilizza in abbinamento con la «JSR».
SBC: esegue una sottrazione tra dato fornito e accumulatore, con riporto.
SEC: setta il flag di carry.
SED: setta il flag D.
SEI: setta il flag I.
STA: simile alla «LDA», ma effettua l'operazione inversa (l'accumulatore viene depositato nella locazione di memoria specificata).
STX: simile alla «STA», ma deposita il registro X.
STY: simile alla «STX», ma deposita il registro Y.
TAX: copia l'accumulatore nel registro X.
TAY: copia l'accumulatore nel registro Y.
TSX: copia lo STACK POINTER nel registro X.
TXA: copia il registro X nell'accumulatore.
TXS: copia il registro X nello stack pointer.
TYA: copia il registro Y nell'accumulatore.

ca che c'é stato il «coinvolgimento» dell'ottavo bit che in queste operazioni, invece, deve essere usato solo come segno. Il successivo flag è quello di break (B): esso viene posto ■ uno dall'istruzione «BRK» per differenziare l'interrupt che essa genera da un interrupt hardware. Il bit 3 (quarto bit) indica che è stato attivato il modo decimale (BCD) ■ che quindi le operazioni vengono fatte seguendo tale metodo (torneremo a discuterne). Il flag I (interrupt) viene condizionato dal programmatore con le istruzioni «SEI» e «CLI» per disabilitare o riabilitare le interruzioni. Infine il flag di carry, che indica un riporto o un prestito e quindi viene attivato (posto a «uno») quando si passa dal valore 255 al valore 0 (con un incremento) e viceversa. Inoltre detto flag viene coinvolto nelle operazioni di «scorrimento» quali «ASL», «LSR», «ROL» e «ROR». Spiegato finalmente il significato e la funzione del registro di stato saranno ora più chiare anche le istruzioni di salto condizionato. C'é da spendere, in ogni modo, ancora qualche parola su queste istruzioni. Esse non effettuano un vero e proprio «salto», nel senso che non possono saltare liberamente a tutte le locazioni

```
:8D97 EA31 11 D0 A9 F0 8D
.8000 78        SEI
.8001 A9 7F     LDA #$7F
.8003 8D 0D DC STA $DC0D
.8006 A9 24     LDA #$24
.8008 A2 80     LDX #$80
.800A 8D 14 03 STA $0314
.800D 8E 15 03 STX $0315
.8010 A9 00     LDA #$00
.8012 8D 12 D0 STA $D012
.8015 AD 11 D0 LDA $D011
.8018 09 80     ORA #$80
.801A 8D 11 D0 STA $D011
.801D A9 01     LDA #$01
.801F 8D 1A D0 STA $D01A
.8022 58        CLI
.8023 60        RTS
.8024 A9 FF     LDA #$FF
.8026 8D 19 D0 STA $D019
.8029 A2 10     LDX #$10
.802B BD 00 81 LDA $8100,X
.802E 9D 00 D0 9TA $D000,X
.8031 CA        DEX
.8032 10 F7     BPL $802B
.8034 A2 07     LDX #$07
.8036 BD 11 81 LDA $8111,X
.8039 9D ■■ 07 STA $07FB,X
.803C CA        DEX
.803D 10 F7     BPL $8036
.803F A2 07     LDX #$07
.8041 BD 20 81 LDA $8120,X
.8044 9D 27 D0 STA $D027,X
.8047 CA        DEX
.8048 10 F7     BPL $8041
.804A A9 57     LDA #$57
.804C A2 80     LDX #$80
.804E 8D 14 03 STA $0314
.8051 8E 15 03 STX $0315
.8054 4C 31 EA JMP $EA31
.8057 A9 FF     LDA #$FF
.8059 8D 19 D0 STA $D019
.805C A2 10     LDX #$10
.805E BD 30 81 LDA $8130,X
.8061 9D 00 D0 STA $D000,X
.8064 CA        DEX
.8065 10 F7     BPL $805E
.8067 A2 07     LDX #$07
.8069 BD 41 81 LDA $8141,X
.806C 9D FB 07 STA $07FB,X
.806F CA        DEX
.8070 10 F7     BPL $8069
.8072 A2 07     LDX #$07
.8074 BD 50 81 LDA $8150,X
.8077 9D 27 D0 STA $D027,X
.807A CA        DEX
.807B 10 F7     BPL $8074
.807D A9 24     LDA #$24
.807F A2 80     LDX #$80
.8081 8D 14 03 STA $0314
.8084 8E 15 03 STX $0315
.8087 4C 31 EA JMP $EA31
.808A EA        NOP
.808B EA        NOP
.808C EA        NOP
.808D EA        NOP
.808E EA        NOP
.808F EA        NOP
```

Figura 4 - Listato del moltiplicatore in alternata

```
:8D97 EA31 11 D0 A9 F0 8D
.9600 A9 17     LDA #$17
.9602 85 FC     STA $FC
.9604 A4 FC     LDY $FC
.9606 A2 00     LDX #$00
.9608 BD 00 95 LDA $9500,X
.960B DD 01 95 CMP $9501,X
.960E 90 0D     BCC $961D
.9610 85 FB     STA $FB
.9612 BD 01 95 LDA $9501,X
.9615 9D 00 95 STA $9500,X
.9618 A5 FB     LDA $FB
.961A 9D 01 95 STA $9501,X
.961D E8        INX
.961E 88        DEY
.961F F0 03     BEQ $9624
.9621 4C 08 96 JMP $9608
.9624 C6 FC     DEC $FC
.9626 A5 FC     LDA $FC
.9628 F0 03     BEQ $962D
.962A 4C 04 96 JMP $9604
.962D 4C 41 96 JMP $9641
.9630 A2 17     LDX #$17
.9632 BD C0 95 LDA $95C0,X
.9635 9D 00 95 STA $9500,X
.9638 9D 60 95 STA $9560,X
.963B CA        DEX
.963C 10 F4     BPL $9632
.963E 4C 00 96 JMP $9600
.9641 A0 00     LDY #$00
.9643 A2 17     LDX #$17
.9645 B9 00 95 LDA $9500,Y
.9648 DD 60 95 CMP $9560,X
.964B F0 03     BEQ $9650
.964D CA        DEX
.964E D0 F8     BNE $9648
.9650 A9 FF     LDA #$FF
.9652 9D 60 95 STA $9560,X
.9655 BD 78 9■ LDA $9578,X
.9658 95 18 95 STA $9518,Y
.965B BD 90 95 LDA $9590,X
.965E 99 30 95 STA $9530,Y
.9661 BD A8 95 LDA $95A8,X
.9664 99 48 95 STA $9548,Y
.9667 C8        INY
.9668 C0 18     CPY #$18
.966A F0 03     BEQ $966F
.966C 4C 43 96 JMP $9643
.966F A9 FF     LDA #$FF
.9671 8D FF 95 STA $95FF
.9674 60        RTS
.9675 EA        NOP
.9676 EA        NOP
.9677 EA        NOP
.9678 EA        NOP
.9679 EA        NOP
.967A EA        NOP
.967B EA        NOP
.967C EA        NOP
.967D EA        NOP
.967E EA        NOP
.967F EA        NOP
.9680 78        SEI
.9681 A9 7F     LDA #$7F
.9683 8D 0D DC STA $DC0D
.9686 A9 80     LDA #$80
.9688 A2 96     LDX #$96
.968A 8D 14 03 STA $0314
.968D 8E 15 03 STX $0315
.9690 A9 00     LDA #$00
.9692 8D 12 D0 STA $D012
.9695 AD 11 D0 LDA $D011
.9698 29 7F     AND #$7F
```

```
.969A 8D 11 D0 STA $D011
.969D A9 01     LDA #$01
.969F 8D 1A D0 STA $D01A
.96A2 58        CLI
.96A3 60        RTS
.96A4 01 02     ORA ($02,X)
.96A6 04        ???
.96A7 08        PHP
.96A8 10 20     BPL $96CA
.96AA 40        RTI
.96AB 80        ???
.96AC EA        NOP
.96AD EA        NOP
.96AE EA        NOP
.96AF EA        NOP
.96B0 A9 FF     LDA #$FF
.96B2 8D 19 D0 STA $D019
.96B5 A9 00     LDA #$00
.96B7 8D 10 D0 STA $D010
.96BA A2 07     LDX #$07
.96BC A0 0E     LDY #$0E
.96BE BD 00 97 LDA $9700,X
.96C1 99 01 D0 STA $D001,Y
.96C4 BD 18 97 LDA $9718,X
.96C7 99 00 D0 STA $D000,Y
.96CA BD 30 97 LDA $9730,X
.96CD F0 09     BEQ $96D8
.96CF AD 10 D0 LDA $D010
.96D2 1D A4 96 ORA $96A4,X
.96D5 8D 10 D0 STA $D010
.96D8 BD 48 97 LDA $9748,X
.96DB 9D F8 07 STA $07F8,X
.96DE 88        DEY
.96DF 88        DEY
.96E0 CA        DEX
.96E1 10 DB     BPL $96BE
.96E3 AD 08 97 LDA $9708
.96E6 E9 05     SBC #$05
.96E8 8D 12 D0 STA $D01.
.96EB A9 60     LDA #$60
.96ED A2 97     LDX #$97
.96EF 8D 14 03 STA $0314
.96F2 8E 15 03 STX $0315
.96F5 4C BC FE JMP $FEBC
.96F8 EA        NOP
.96F9 EA        NOP
.96FA EA        NOP
.96FB EA        NOP
.96FC EA        NOP
.96FD EA        NOP
.96FE EA        NOP
.96FF EA        NOP

.9760 A9 FF     LDA #$FF
.9762 8D 19 D0 STA $D019
.9765 A9 00     LDA #$00
.9767 8D 10 D0 STA $D010
.976A A2 07     LDX #$07
.976C A0 0E     LDY #$0E
.976E BD 08 97 LDA $9708,X
.9771 99 01 D0 STA $D001,Y
.9774 BD 20 97 LDA $9720,X
.9777 99 00 D0 STA $D000,Y
.977A BD 38 97 LDA $9738,X
.977D F0 09     BEQ $9788
.977F AD 10 D0 LDA $D010
.9782 1D A4 96 ORA $96A4,X
.9785 8D 10 D0 STA $D010
.9788 BD 50 97 LDA $9750,X
.978B 9D F8 07 STA $07F8,X
```

```
.978E 88        DEY
.978F 88        DEY
.9790 CA        DEX
.9791 10 DB     BPL $976E
.9793 AD 10 97 LDA $9710
.9796 E9 00     SBC #$00
.9798 8D 00 D0 STA $D000
.979B A9 A8     LDA #$08
.979D A2 97     LDX #$97
.979F 8D 14 03 STA $0314
.97A2 8E 15 03 STX $0315
.97A5 4C BC FE JMP $FEBC
.97A8 A9 FF     LDA #$FF
.97AA 8D 19 D0 STA $D019
.97AD A9 00     LDA #$00
.97AF 8D 10 D0 STA $D010
.97B2 A2 07     LDX #$07
.97B4 A0 0E     LDY #$0E
.97B6 BD 10 97 LDA $9710,X
.97B9 99 01 D0 STA $D001,Y
.97BC BD 28 97 LDA $9728,X
.97BF 99 00 D0 STA $D000,Y
.97C2 BD 40 97 LDA $9740,X
.97C5 F0 09     BEQ $97D0
.97C7 AD 10 D0 LDA $D010
.97CA 1D A4 96 ORA $96A4,X
.97CD 8D 10 D0 STA $D010
.97D0 BD 58 97 LDA $9758,X
.97D3 9D F8 07 STA $07F8,X
.97D6 88        DEY
.97D7 88        DEY
.97D8 CA        DEX
.97D9 10 DB     BPL $97B6
.97DB A9 00     LDA #$00
.97DD 8D 12 D0 STA $D012
.97E0 AD 11 D0 LDA $D011
.97E3 09 80     ORA #$80
.97E5 8D 11 D0 STA $D011
.97E8 A9 F5     LDA #$F5
.97EA A2 97     LDX #$97
.97EC 8D 14 03 STA $0314
.97EF 8E 15 03 STX $0315
.97F2 4C BC FE JMP $FEBC
.97F5 A9 FF     LDA #$FF
.97F7 8D 19 D0 STA $D019
.97FA AD FF 95 LDA $95FF
.97FD F0 0B     BED $980A
.97FF A2 5F     LDX #$5F
.9801 BD 60 95 LDA $9560,X
.9804 9D 00 97 STA $9700,X
.9807 CA        DEX
.9808 10 F7     BPL $9801
.980A A9 00     LDA #$00
.980C 8D 12 D0 STA $D012
.980F AD 11 D0 LDA $D011
.9812 29 7F     AND #$7F
.9814 8D 11 D0 STA $D011
.9817 A9 80     LDA #$80
.9819 A2 96     LDX #$96
.981B 8D 14 03 STA $0314
.981E 8E 15 03 STX $0315
.9821 A9 00     LDA #$00
.9823 8D FF 95 STA $95FF
.9826 8D 10 D0 STA $D010
.9829 4C 31 EA JMP $EA31
.982C EA        NOP
.982D EA        NOP
.982E EA        NOP
.982F EA        NOP
```

Figura 5 - Listato del moltiplicatore riordinatore

I nuovi registri dei 24 sprite

Coordinate Y	Coordinate X	Coordinate X (MSB)	Forme
$9500-$9517	$9518-$952F	$9530-$9547	$9548-$955F

Figura 6

di memoria possibili. Il loro salto è del tipo «relativo», ovvero è in funzione della posizione che esse occupano in memoria. Se difatti ci andiamo a riguardare il vecchio listato del moltiplicatore di sprite (MC n. 73), possiamo osservare che nella colonna relativa ai codici LM esadecimali (quella centrale), in corrispondenza di ciascuna «BPL», troviamo i valori $10 ed $F7. Il primo si riferisce al codice operativo, mentre il secondo è lo «scostamento» in complemento a due. Tradotto in altre parole, le istruzioni di salto condizionato possono effettuare spostamenti in avanti e indietro limitati ai valori +127 e −128. Il valore esadecimale $F7 corrisponde al valore (usando il complemento a due) −8. Ciò si traduce in un salto all'indietro di 8 locazioni di memoria. Le tre «BPL» delle rispettive tre subroutine hanno tutte uno scostamento di «−8» proprio perché devono tutte saltare di 8 locazioni indietro dal punto in cui si trovano! Non è necessario tuttavia, in fase di programmazione, calcolare l'esatta locazione di scostamento «a mano», in quanto praticamente tutti gli assemblatori effettuano detta operazione automaticamente (basta indicare la locazione di memoria alla quale vogliamo saltare). L'unico accorgimento da osservare riguarda appunto i «limiti» di salto che non devono essere oltrepassati, pena un «rifiuto di assemblaggio» da parte del nostro MONITOR-ASSEMBLATORE.

Del vecchio listato non resta da spiegare altro se non il significato delle «JMP $FEBC» e della «JMP $EA31». Dalla locazione $FEBC in poi troviamo una routine del sistema operativo che serve a ripristinare i valori dei registri X e Y e dell'accumulatore con i valori da questi memorizzati prima che avvenisse la IRQ e a ritornare al punto esatto in cui si era interrotto il programma principale. Dalla locazione $EA31 in poi troviamo invece la routine originale delle IRQ provocate dal timer. Essendo quest'ultima routine eseguita normalmente una volta ogni sessantesimo di secondo mentre le tre IRQ avvengono nel giro di un cinquantesimo di secondo, è opportuno, per non variare troppo il tempo di chiamata di detta routine, eseguirla una volta su tre.

E ora veniamo alle «tecniche alternative» (finalmente!).

La prima tecnica, fattami tornare a mente da un lettore di Roma, Massimiliano Leoni, consiste nel «duplicare» gli sprite alternando, con due gruppi di valori, i loro attributi ogni qualvolta si verifica il completamento del tragitto del pennello elettronico, quindi gli sprite vengono riutilizzati in «pennellate alterne». Il listato che realizza quanto detto è quello di figura 4. Il primo blocco di istruzioni (molto simile a quello già utilizzato per l'altro listato) si occupa della definizione di una nuova routine IRQ, che verrà attivata ogni qualvolta il raster arriverà alla posizione «256». Detta routine effettua un trasferimento delle coordinate, delle forme e dei colori, da tre tabelle ai relativi registri degli otto sprite e imposta l'indirizzo di una nuova routine IRQ nelle locazioni $0314 e $0315. Al verificarsi della nuova IRQ ci sarà un nuovo trasferimento, simile al precedente, ma questa volta le tabelle saranno diverse. Ci sarà inoltre il ripristino della routine IRQ che aveva abilitato quella attualmente in esecuzione (quindi la precedente) e il ciclo si ripeterà dall'inizio. Con questo sistema gli sprite sono liberi di muoversi lungo tutta l'area dello schermo, ma com'è facile intuire, «sfarfalleggiano» in modo visibilissimo e, tra l'altro, diventano anche trasparenti. Il trucco in pratica cerca di sfruttare il fenomeno della «persistenza delle immagini», ma ci riesce molto male. Meglio quindi avere una limitazione nel posizionamento degli sprite piuttosto che il fastidiosissimo effetto «fantasma-sfarfallante».

La seconda tecnica si avvale sempre della «riutilizzazione degli sprite nella stessa pennellata video» (come faceva il listato del numero 73), ma in modo un po' più avanzato. Il listato che realizza ciò è quello di figura 5. Esso permette di visualizzare 24 sprite posizionabili liberamente su tutto lo schermo, ammesso che non ve ne siano più di otto con la «stessa» coordinata Y. Per farlo funzionare è necessario impostare una:

```
SYS 38528
```

Detto questo passiamo ad esaminare il principio di funzionamento. Il succo di tutto il programma sta nella routine che effettua il «riordinamento degli sprite», che consiste nel mettere in ordine crescente le coordinate Y degli sprite e in seguito di associare ad ogni coordinata Y gli altri attributi dello sprite corrispondente, quindi effettuando un riordinamento anche di questi. Così facendo avremo da una parte l'elenco originario dei nostri 24 sprite, che utilizzeremo come se fosse un gruppo di registri di I/O, e dall'altra un elenco in ordine «crescente di coordinata Y», che verrà utilizzato dalla routine moltiplicatrice funzionante in IRQ. Che cosa se ne fa la routine moltiplicatrice dell'elenco ordinato? Semplice: imposta tre diverse IRQ raster, ognuna corrispondente alla coordinata Y rispettivamente del primo, del nono e del diciassettesimo sprite. Per quello che riguarda la prima IRQ essa può avvenire anche alla posizione «zero» del pennello elettronico; le altre due avverranno un po' prima delle effettive coordinate degli sprite (a causa dei ritardi). Al verificarsi della prima IRQ, la routine abilitata esegue un trasferimento di dati degli attributi dei primi otto sprite, dalla tabella ordinata ai registri degli otto sprite «originali» e imposta la nuova posizione del raster alla quale avverrà la prossima interrupt. Inoltre effettua un cambiamento della routine IRQ. La prossima IRQ quindi «chiamerà» una nuova routine che questa volta effettuerà un trasferimento dei dati ordinati del secondo gruppo di otto sprite e, come la prima, muterà l'indirizzo della routine IRQ e la posizione raster alla quale avverrà l'interruzione. La terza routine, chiaramente, effettuerà l'ultimo trasferimento relativo al terzo gruppo di otto sprite. Verrà di seguito abilitata una ultima routine, la quale si occupa della comunicazione tra la routine riordinatrice (che non funziona in IRQ) e la routine moltiplicatrice. Questa routine di interfaccia è necessaria in quanto la routine di riordinamento impiega una frazione di tempo non trascurabile per eseguire la sua operazione e quindi se funzionasse in sincronismo con il raster ruberebbe troppo tempo al resto del programma. Così facendo, invece, la routine riordinatrice può essere chiamata quando vogliamo (ovvero ogni qualvolta c'è un aggiornamento, seppur minimo, delle caratteristiche dei 24 sprite), ma il suo «intralcio» sarà notevolmente ridotto. La comunicazione avviene tramite un byte. Quando la routine riordinatrice ha finito il suo compito essa pone a $FF il byte $95FF. Questo abilita la routine di interfaccia che effettua una copia della tabella ordinata, sicura che in questo momento è stata effettivamente resa tale, e questa copia sarà quella che verrà usata dal resto della routine IRQ. Inoltre detta routine imposterà a zero il menzionato byte $95FF avvisando quindi l'utente che è possibile effettuare un

nuovo riordinamento. A questo punto verrà riabilitata la prima routine IRQ e tutto ricomincerà da capo.

Osserviamo ora come viene effettuato il riordinamento. Innanzi tutto è necessaria una doppia copia delle coordinate Y. Una servirà per il riordinamento delle stesse e l'altra per riordinare anche il resto. Dette copie avvengono chiamando la routine posizionata a partire da $9630 con una

SYS 38448

Effettuata questa operazione preliminare il programma devierà alla locazione $9600 dove troviamo l'algoritmo di riordinamento. Spieghiamone il funzionamento. I 24 numeri, rappresentanti le coordinate Y di altrettanti sprite, sono disposti in ordine casuale. Il programma prende il primo numero della lista e lo confronta con il secondo. Se in detto confronto il primo numero risulta più grande del secondo il programma effettua uno scambio di locazione tra i due, altrimenti lascia inalterate le loro posizioni. Si passa quindi a confrontare il secondo con il terzo numero. Anche qui vale la stessa legge di prima: «scambio» se il secondo è più grande del terzo e altrimenti «posizioni inalterate». Si prosegue così fino a lista terminata. A questo punto il numero più grande della lista si ritroverà sicuramente per ultimo (provate con degli esempi pratici e vi convincerete). La lista comunque non è ancora ordinata. Si effettua quindi di nuovo lo stesso procedimento di «scambio/non scambio» dall'inizio della lista, ma questa volta tralasciando l'ultimo confronto (quello tra il ventitreesimo e il ventiquattresimo numero), dato che il più grande tra i due si trova già nella giusta posizione. Al termine di questa seconda «passata» il secondo numero più grande si ritroverà nella posizione 23. Ancora un'altra passata (tralasciando il confronto 22/23) e anche il terzo numero più grande si ritroverà nella sua giusta posizione (ovviamente la 22). Si procede così fin quando sarà possibile tralasciare anche lo scambio tra il primo e il secondo numero (nota di Andrea de Prisco: a dire il vero la lista risulta ordinata non appena si esegue una passata senza effettuare scambi, e ciò può succedere anche prima dell'ultimo «giro». Tale tecnica, nota in Informatica, si chiama «metodo del gorgogliamento»). A questo punto la lista sarà completamente ordinata. Questo apparentemente complesso sistema utilizzato è (strano a dirsi) quello che ci permette il riordinamento più rapido possibile (almeno spero...). La copia che inizialmente avevamo fatto si è trasformata (a meno che le coordinate non fossero già tutte in ordine) e quindi è servita per non intaccare la lista originale (quella di figura 6), che chiaramente deve restare inalterata. La seconda copia ci serve ora che il programma salta alla locazione $9641. Questa nuova routine si occupa del già citato riordinamento del «resto degli attributi». Dunque, prendiamo la prima coordinata Y dalla lista ordinata e andiamo a ricercare la sua posizione equivalente nella lista non ordinata (non quella originale, ma quella della seconda copia). Quando la troviamo possiamo cancellare (con un $FF), dalla copia della lista non ordinata, la coordinata Y, ma nello stesso tempo eseguiamo un trasferimento, dalla lista originale alla equivalente posizione ordinata (in questo caso la prima posizione), della coordinata X e della forma del corrispondente sprite. Il valore $FF, usato come «cancellino», serve ad eliminare dalla prossima ricerca i dati relativi a questo sprite, perché ormai sono già stati riordinati. È opportuno quindi non utilizzare come coordinata Y la posizione $FF in quanto altrimenti si creerebbe confusione tra dati riordinati e non (tra l'altro il valore $FF posizionerebbe lo sprite fuori schermo, quindi non serve). Terminata l'esecuzione di quest'ultima routine il programma impone, come detto, la locazione $95FF a $FF e in seguito ritorna al programma principale (che poi sarebbe il gioco). A questo punto, a partire dalla locazione $9500 fino alla $955F abbiamo gli attributi degli sprite completamente ordinati e quando avverrà l'IRQ alla posizione raster 256 l'apposita routine potrà usufruire dei nuovi dati. I più attenti avranno notato che i colori non vengono utilizzati, ma questa apparente limitazione non è difficile da eliminare (... lascio a voi il divertimento).

Esistono delle limitazioni nel posizionamento degli sprite che derivano dal fatto che i 24 sprite devono essere posizionati in tre «blocchi» distinti. La prima è quella che, chiaramente, non è possibile posizionare più di otto sprite «fuori schermo» (cioè nel bordo superiore o inferiore... ai lati invece SI!). L'altra invece è che la routine non controlla automaticamente l'eventualità che più di otto sprite siano posizionati nello stesso «blocco» e quindi si «semi-blocca» se accade un avvenimento del genere, ovvero rallenta notevolmente l'esecuzione del programma, generando tra l'altro anche lo sfarfallamento degli sprite. Entrambe le limitazioni, tuttavia, sono eliminabili senza andare incontro a grossi problemi e i più bravi di voi (cioè i più interessati) sapranno certamente cavarsela da soli. In ogni modo, se ne farete richiesta, cercherò di accontentarvi. Buone vacanze!

Megaposta

Siamo due assidui lettori di questa bella rivista, in particolare seguiamo con vera trepidazione l'evolversi di questa sua iniziativa, che ci pare particolarmente interessante e azzeccata. Se scriviamo è per dare il nostro contributo convinti che la nostra esperienza possa essere di spunto ad altri intraprendenti. Ma passiamo adesso al dunque:

Abbiamo affrontato la situazione in maniera tecnica cercando dapprima di rimediare alle apparenti carenze strutturali del nostro amato Commodore 64. Infatti, come tutti ben sanno, la mancanza di hardware grafico limita a poche decine il numero di poligoni pieni tracciabili per secondo...

...Se lei desidera possiamo inviarle un dischetto contenente un dimostrativo delle suddette tecniche.

Tommaso e Luigi Bini, Pistoia.

Innanzi tutto mi scuso con voi per il ritardo con cui questa lettera appare sulle pagine di MC e del grande «taglio centrale» che ho dovuto apportare. Speriamo che nel frattempo non abbiate cambiato idea; sono molto curioso di vedere cosa siete riusciti a combinare!

Ciao Marco, voglio essere molto breve e dirti la ragione di questa lettera. Sto seguendo con molta curiosità i tuoi articoli e lo svolgersi del «Megagame 64». Ho capito che oltre al mitico 64 sei attratto in modo morboso dall'Amiga. Bene, premesso che sono un possessore felice di entrambi i computer e che ovviamente mi sento attratto dalla grafica dei 512 K ho pensato: perché non fare un videogioco per Amiga veloce, ben strutturato, bello graficamente e che ti prenda da morire come OUT RUN?...

Michele Signorile, Bari

Anche tu scusami per il «mutilamento», ma anche così il gioco farebbe una bellissima figura (senza andare a realizzare costose interfacce)...

Inoltre avverto Paolo Costabel, Roberto Sorgente e Marco Marinai che ho letto le loro lettere (anche quella di Marco su dischetto) ma non ho il tempo di rispondergli... alla prossima puntata.

...E infine un grazie a Raffaele Cirillo (Napoli) e a Tonino Librici (Torino) per aver inviato il loro contributo.

di Marco Pesce

Un videogioco tutto nostro

nona puntata

Coraggio, non fate quelle facce ... in fondo sono state delle belle vacanze, ma come ogni cosa bella... si torna al lavoro!
Permettetemi di ricordare a tutti coloro che dicono: «ma allora, quando lo finite ...» che la rubrica Megagame 64 è stata istituita per dare la possibilità a tutti i programmatori italiani di mettersi in luce e non per dimostrare che Marco Pesce sa programmare videogiochi, quindi non è il caso di metterci tutta questa fretta, soprattutto considerando che con il sistema che stiamo utilizzando i tempi di realizzazione vengono quadruplicati e che «normalmente» per realizzare un buon gioco sono necessari 8/9 mesi (a dir poco)

Piuttosto che realizzare la solita accozzaglia di istruzioni è preferibile spendere un po' più di tempo per curare i dettagli teorici e quindi evitare di cominciare a programmare routine che poi dovranno essere notevolmente modificate o addirittura sostituite con tutti i relativi problemi che questa operazione comporta di solito; una volta strutturato nei minimi dettagli il lavoro di programmazione sarà facilitato ■ quindi più rapido da eseguire ■ gli intoppi saranno ridotti all'osso (... almeno spero).

Questo mese ci dedichiamo all'ascolto dei suggerimenti inviati da un nostro lettore, Marco Marinai, che, al momento in cui scrivo, è l'unico che si trova al passo con i tempi (molti continuano ancora a mandarmi idee per nuovi megagame ...). Secondo lui il modo migliore per far notare il nostro videogioco ■ quello di dotarlo di una grafica eccezionale (in modo da renderlo competitivo con i prodotti di Amiga ■ compagnia bella). Indubbiamente il fattore più importante in un videogioco è la grafica ■ ciò è pienamente giustificato dal fatto che il senso più evoluto dell'uomo è proprio la vista. Tuttavia, come molti cominciano ■ capire, la grafica non è tutto (com'è vecchio questo discorso) .. deve anche esserci una «facile interazione». Con questo il nostro Marco vuol dire che il gioco deve essere utilizzabile senza l'aiuto di manuali enciclopedici; in ogni modo anche un giocatore esperto dovrà essere soddisfatto da nuove difficoltà, man mano che si addentra nel gioco. Altro elemento di fondamentale importanza: il suono. Si consiglia l'utilizzo dei suoni digitalizzati durante il gioco ■ di una buona musica introduttiva.

Non ultima l'ampiezza dell'area di gioco: una grande mappa da esplorare è disegnata in modo tale da «incuriosire il giocatore».

Infine l'utilizzo di un modello di gioco già esistente ma senza fare il «copiato»: Shadowfire (vedi foto 1, 2, 3 e 4).

Cominciamo col dire che sono pienamente d'accordo con Marco sul fatto che la grafica deve essere di ottima qualità; infatti non vedo perché bisognerebbe disegnare in modo pietoso quando con la stessa capacità di memoria si potrebbero ottenere risultati molto migliori. Qualcuno è convinto che la grafica sia solo un pretesto per attirare l'attenzione dell'acquirente ed in molti casi è vero, ad esempio per le avventure, che si propongono di essere libri interattivi e in quanto tali non necessitano indispensabilmente di figure. In un gioco arcade tuttavia non ha senso disegnare sprite in modo rozzo quando è possibile fare diversamente, e ciò diventa inevitabil-

Foto 1

Foto 2

forze armate ecc.). Infine nel riquadro centrale viene visualizzato l'interno del centro in cui siamo, con tanto di animazioni varie. Da notare che la schermata descritta finora è quella principale ■ che essa verrà sostituita nel caso di attacchi, esplorazioni etc. con una visione a pieno schermo dell'azione.

La suddivisione particolare dello schermo ci permette di realizzare uno scrolling per quanto riguarda la sezione del tunnel, tuttavia ciò non è realizzabile anche per il riquadro centrale che riprende la scena del compartimento. In quest'ultimo caso dovremo accontentarci di una schermata fissa oppure di uno scrolling tipo «The Sentinel», cioè che avviene ad alta velocità (otto pixel per quadro)

mente indice di trascuratezza. Anche in questi casi però ■ volte si eccede, in quanto pur di ottenere delle belle visuali si sacrifica tutto il resto e il risultato è tanto fumo negli occhi e nient'altro. Resta chiaro comunque che un gioco con una bella grafica non è necessariamente una presa in giro e che non è mai saggio giudicare dalla prima impressione. Per realizzare la parte grafica Marco consiglia «l'utilizzo» di un bravo disegnatore (lui si esclude ...) ■ di un bravo animatore e progettista di routine grafiche (e qui propone il lavoro al sottoscritto e a chiunque sia in grado di aiutarlo ...). Considerando le icone di Shadowfire a volte un po' indistinte sarebbe opportuno lavorare in alta risoluzione per ottenere dei buoni risultati. In ogni modo la parte relativa all'azione dinamica può essere disegnata in uno qualsiasi dei modi grafici.

Considerando la sezione riguardante l'interazione ci viene proposto uno schema che io ho tradotto in schermate (foto 5 e 6). Lo schermo è diviso in più parti. Il tunnel tridimensionale è stato sostituito con uno ■ due dimensioni, più facile da realizzare ma anche più rapido da gestire (in ogni modo, molto elegante) visibile nella parte bassa dello schermo (foto 5) che viene attivato solo quanto dobbiamo eseguire i tanto famosi spostamenti, altrimenti sostituito dall'interfaccia ad icone per utilizzare le apparecchiature del settore in cui ci troviamo (foto 6). La parte di schermo in alto a sinistra contiene il computer tascabile che ci informa di eventuali allarmi o cose del genere e funziona come una telescrivente (... alla faccia dell'originalità). Infine nel riquadro a destra troviamo un pannello che contiene le informazioni più importanti e che quindi devono essere sempre visibili (condizioni economiche, quantità delle

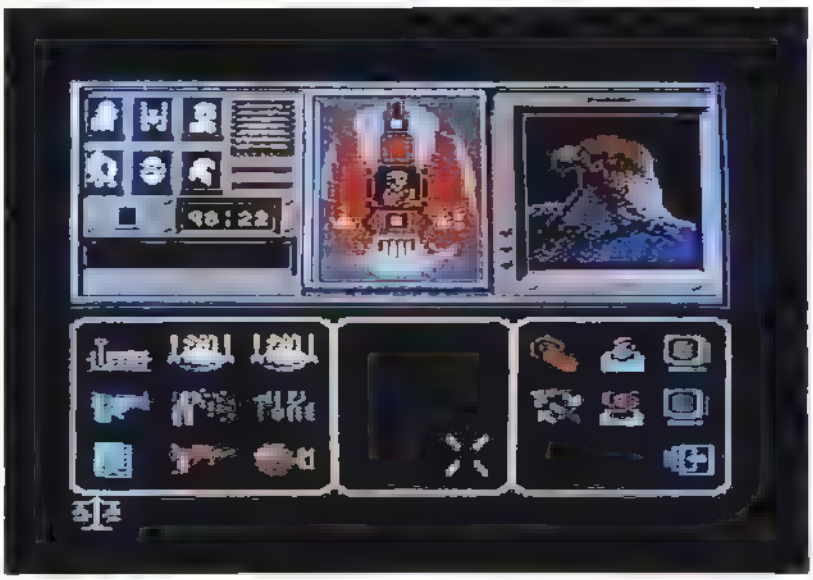


Foto 3

Foto 4

ma solo quando usciamo da un lato dello schermo. Per il resto lo schema risulta né troppo pieno di gadget né troppo ... vuoto (entrambe le situazioni metterebbero a disagio il giocatore inesperto).

Passiamo all'audio. Si ribadisce l'importanza di questo elemento che in molti casi si rivela un ottimo stimolo per continuare a giocare. Altre volte invece, quando è mal realizzato, provoca addirittura fastidio (... meglio non fare nomi). Da parte mia penso che oltre a dei buoni effetti sonori è necessaria anche una colonna sonora interattiva, cioè che si adatta alla particolare azione del momento e che ricrea quindi la giusta «atmosfera». I suoni digitalizzati indubbiamente sono di grande effetto, ma in termini di memoria le cose vanno piuttosto maluccio. Tuttavia non è escluso che qualcosa si possa fare, almeno per quello che riguarda le situazioni più importanti.

Per quanto riguarda l'estensione del campo di gioco Marco suggerisce di dedicargli una buona fetta di RAM e quindi di assicurarne la vastità.

Detto questo non posso far altra cosa se non sperare che anche qualcun altro abbia avuto la stessa idea del Marinai e che gli eventuali nuovi suggerimenti siano altrettanto validi.

◄ Foto 5

Foto 6 77 ►

Scuola di videogame

Suoni digitalizzati

Visto che si è tanto discusso di sonoro e effetti digitalizzati perché non parlarne anche dal punto di vista tecnico?

Il primo problema è: «da dove cominciare ...». Su questo argomento si è molto discusso (anche sulle pagine di MC) e molti di voi probabilmente già sanno di cosa si tratta. Altrettanti di voi, sempre probabilmente, sanno già quali sono i problemi collegati all'uso di questa tecnica sul Commodore 64. Infine altri, sicuramente, non ne sanno assolutamente niente ... Come fare per accontentare tutte e tre le «categorie»? ... Cominciamo da zero! Partendo dalla definizione di suono vera e propria possiamo dire che esso è prodotto dalla vibrazione meccanica di un corpo che se avviene in un determinato campo di frequenze stimola l'organo dell'udito. Questa vibrazione produce un suono diverso a seconda della forma del corpo sottoposto a vibrazione dell'intensità di questa vibrazione che varia in funzione della quantità di energia impressa al corpo per porlo in vibrazione. In parole più spicce, se provochiamo un urto tra due metalli il suono dipenderà anche dalla forza che impieghiamo per compiere questa operazione. Il sistema per riprodurre un suono generato da una qualsiasi vibrazione è quello di copiarne la «forma d'onda», ovvero tutti gli «spostamenti d'aria» che esso ha provocato sull'organo dell'udito. Dette vibrazioni possono essere memorizzate e quindi riprodotte da un apposito apparecchio quale un altoparlante. Un normale registratore audio memorizza questa forma d'onda in modo analogico, ovvero, se non ci fossero i disturbi, nella forma più perfetta possibile. Tuttavia i disturbi ci sono e quindi si preferisce a volte il metodo digitale, che pur presentando una certa approssimazione permette di ridurli in modo notevole. Ci siamo allontanati molto da quello che è l'argomento principale, ma ora ci ritorniamo subito. Il C-64 essendo un computer non può che adottare la forma digitale. Fortunatamente esso è dotato di un convertitore digitale-analogico, che ha la funzione di trasformare un determinato valore digitale in un altrettanto determinato spostamento della membrana di un altoparlante. La risoluzione di detto convertitore è di 4 bit. Ciò vuol dire che sono possibili, purtroppo, solo 16 diverse posizioni della membrana dell'altoparlante e chiaramente il tutto va a scapito della qualità dei suoni riprodotti. Tuttavia noi non abbiamo nessuna pretesa di alta fedeltà (almeno spero) e quindi possiamo accontentarci. Ma un altro problema ci taglia le gambe: la quantità di memoria disponibile. Infatti volendo generare un suono campionato alla frequenza di 6400 hertz (... per far cifra tonda) con 64 K di memoria il tutto dura solo 20 secondi, che in fondo non

sarebbero neanche tanto pochi, ma normalmente ci occorre anche molta memoria per il gioco ■ quindi non resta molto spazio per questo genere di suoni. Ultima «mazzata»: il tempo. Premessa: il convertitore digitale-analogico del Commodore-64 non funziona autonomamente, ovvero non prende «da solo» i dati della memoria. Questo vuol dire che sarà il microprocessore a doverglieli fornire, e vuol anche dire che il tempo necessario a far ciò sarà rubato al resto del programma. Facciamo un po' di conti. 6400 hertz divisi per 50 (cicli di schermo al secondo) fanno 128 variazioni dell'altoparlante da effettuare ad ogni scansione del pennello elettronico. Se il nostro codice non occupa troppo tempo e ci lascia quindi lo spazio per fare ciò, siamo veramente fortunati, ma nel caso di giochi che usano in modo intensivo lo scrolling ... beh le cose non vanno molto bene. Certo è possibile diminuire la frequenza del suono, ma chiaramente i risultati peggiorano. È anche possibile far in modo che i suoni siano generati quando non ci sono animazioni troppo complesse in esecuzione e questa, nella maggior parte dei casi, è la soluzione migliore.

Abbiamo finora parlato come se in memoria avessimo già la forma d'onda del nostro suono bella e pronta, ma in realtà questa si deve ottenere tramite l'utilizzo dei tanto famosi digitalizzatori. Queste apparecchiature hardware contengono un convertitore analogico-digitale che praticamente trasforma la variazione continua del segnale in ingresso (suono) in una a «scatti». Nel caso del Commodore-64 gli scatti possibili come detto sono 16. Molto spesso il segnale convertito si invia a una delle porte joystick, collegando i 4 bit di risoluzione del convertitore con i rispettivi piedini del connettore che corrispondono alle quattro posizioni fondamentali del joystick. Fortunatamente la «game port» accetta variazioni anche molto rapide dello stato di detti ingressi e ciò ci permette appunto di utilizzarla anche per il nostro scopo. Ho realizzato qualche giorno fa un digitalizzatore hardware (tempo ancora prima ne avevo realizzato uno molto più «sempliciotto» ...) a otto bit che quindi va bene anche per l'Amiga. Funziona abbastanza bene e credo che il prossimo mese lo presenterò su queste pagine (per la gioia di chi si diverte anche con le interfacce elettroniche), corredato dell'opportuno software che per l'appunto esamineremo la prossima volta.

Megaposta

«Caro Marco,
grazie. Grazie di cuore per aver ascoltato le mie e altrui critiche. Ciò che dici nella sesta puntata del «MEGAGAME» dimostra non solo che sei una di quelle rare persone che sa ammettere i propri errori, ma anche che il tuo interesse per l'arte (posso chiamarla così?) del videogioco va al di là del semplice hobby.

Sono completamente d'accordo con te sulla superiorità del videogame astratto, fatto di stragegia e fantasia, sulla simulazione più o meno banale del reale. Chi va cianciando sulla «simulazione totale» e si augura l'avvento di computer sempre più potenti non ha capito che non sono il numero di pixel o i MIPS che trasformano un ammasso di chip e circuiti stampati in un universo affascinante e magico. L'imitazione della realtà ha il solo effetto di mostrare i limiti del computer, non le sue potenzialità. Nessun simulatore di volo potrà mai competere con un vero aereo in quanto a realismo, nessun clone di Pole Position potrà farci vivere l'emozione di guidare una Ferrari, nessuno sport computerizzato svilupperà i nostri muscoli. Vivere la realtà simulata del computer anziché quella vera è un ripiego, una ammissione della nostra mediocrità. Come tutti i media che imitano la realtà (cinema, televisione etc.) il videogioco di simulazione ci fornisce emozioni di seconda mano, esperienze a basso costo in sostituzione di quelle reali che non possiamo permetterci. C'è chi vive la storia d'amore romantica che non riesce a trovare nella realtà attraverso una telenovela, chi scarica le sue frustazioni di perdente identificandosi nel Rocky vincitore di tutte le sfide, chi si compra Ferrari Formula One come surrogato della Testarossa che non potrà mai permettersi

Se fosse questo il mondo dei videogame e del computer in generale, sarei il primo ■ considerare questo nuovo media come un rimbecillitore di masse, alla stregua della televisione. Ma, come fai giustamente notare, il computer può offrire ben altro. Ci sono infatti esperienze che la realtà non è in grado di farci provare. Quale altro media sarebbe capace di farci identificare in una biglia che rotola in un paesaggio surrealista popolato di amebe e strani esseri, o in un ranocchio che deve attraversare una strada, o in un esserino nasuto che saltella su una piramide di cubi sospesa nel nulla? È questo il tipo di esperienze che solo un computer può farci vivere, è questo l'universo assurdo e irreale che è racchiuso dietro lo schermo del monitor, ed è proprio questo il motivo per cui tanta gente si innamora di queste strane macchine che si chiamano computer ...».

Paolo Costabel, Genova

Che lettera lunga! ma ne è valsa la pena. Ben detto Paolo. Staremo a sentire la replica degli ... altri.

«Spett.le Marco Pesce,
ho letto con interesse quasi tutte le puntate del Magagame e in quella di maggio ho notato lo sfogo contro la ripetitività e la scarsa originalità di idee circa la creazione di un nuovo gioco, ■ a questo proposito ti propongo un gioco che potrebbe piacerti ...».

Roberto S., Roma

Come idea non è male, ma ormai il gioco lo abbiamo cominciato e quindi per ora non possiamo realizzarla ... Scusami per il grande «taglio».

Spettabile Ragioniere (??? n.d.r.),
seguo con molto interesse lo sviluppo del tanto agognato Megagame e devo dire che non sono affatto soddisfatto delle scelte da voi effettuate. Io mi chiedo: perché realizzare un wargame spaziale tipo Defender of the Crown ampliato?

Così facendo non risponderete affatto alle esigenze dell'enorme mondo degli "smanettoni", bensì alla ristretta cerchia di chi ama giochi come Driller, DOTC, ecc.

Cosa chiedo io? Bene, perché non realizzare un gioco ricalcando lo stile di Rastan, Ghosts'n Goblins e company? ...»

Matteo Bittanti,
San Pietro all'Olmo (MI)

Caro Matteo, sei giustificato solo dal fatto che la tua lettera è arrivata prima dell'uscita del numero di luglio/agosto ...

di Marco Pesce

Un videogioco tutto nostro

Mi fa piacere che tra i lettori che seguono lo svolgimento del megagame ce ne siano alcuni particolarmente attivi, ma mi fa anche arrabbiare che gli altri se ne stiano lì per conto loro lanciando una pietruzza ogni tanto (... e finì sommerso dalle meteore). In molti si sono fatti sentire una sola volta e poi sono spariti (forse offesi da come li ho trattati?). Così non va bene

Questo non vuol dire tuttavia che non ci sono i presupposti per continuare il gioco, anzi. Solo che un po' di pareri in più non guasterebbero. Le lettere che mi sono pervenute finora contengono tutte dei consigli su alcuni aspetti del gioco mentre qui ci vuole una dettagliata attenzione ai particolari di «tutto» il gioco. Ciò non vuol dire che ogni lettore doveva inviare una particolareggiata bozza, ma se veramente ogni lettore avesse mandato la sua idea... la bozza dettagliata veniva fuori! Questo per ribadire il concetto che il gioco deve venir fuori dalla collaborazione di tutti i lettori. Con i consigli che ho a disposizione dovrei realizzare la maggior parte del lavoro per conto mio e sinceramente la cosa non è molto sportiva. «L'attivo» Marco Marinai si lamenta per il fatto che i progressi sono stati pochissimi e dice «come potrei aiutarti se non so cosa chiedi veramente». Nessuno ha ancora chiesto qualcosa di particolare, ma forse sono stato poco chiaro io e allora mi scuso con tutti i lettori. Il gioco non è stato affatto definito e ciò viene dimostrato dalle lettere che mi mandate, che contengono aggiunte e modifiche di ogni tipo. Voglio la collaborazione di tutti! E sono sicuro che fra tutte le idee che arriveranno la bozza teorica si potrà completare in ogni sua minima parte. Sarà meglio ricapitolare quanto è stato deciso finora.

Il gioco non è più per un solo giocatore; i contendenti sono diventati necessariamente due (questa modifica renderà il gioco molto più divertente).

Il centro di controllo non esiste più. Esistono invece due astronavi al comando delle quali troviamo ciascuno dei due giocatori. Dette astronavi sono dotate di tutti i mezzi necessari per trasmettere ordini ai pianeti in possesso del loro comandante. I vari centri di ricerca e simili vengono costruiti sui pianeti. Gli spostamenti nello spazio vengono fatti avvalendosi della velocità del famoso Iperspazio. Le trasmissioni dei dati tuttavia vengono fatte con tecniche molto sofisticate che permettono di ridurre a zero i tempi necessari all'operazione. Nella galassia esistono solo due civiltà, ognuna sotto il comando di un giocatore. Ogni giocatore possiede inizialmente il 50% dei pianeti appartenenti alla galassia, quindi viene eliminata la possibilità di esplorazioni e di civiltà neutrali (ciò non è inverosimile... immaginate che le innumerevoli battaglie in questa galassia abbiano comportato la divisione perfetta in due grandi civiltà e che ormai ogni angolo di essa è stato esplorato; il compito di definire l'unica civiltà suprema è affidato ai due comandanti!).

Ogni pianeta possiede le seguenti caratteristiche:
— *Tipo*: (1) abitabile; (2) adatto solo per installazioni minerarie energetiche o militari; (3) adatto solo per l'installazione di basi militari.
— *Quantità di risorse possedute*: (1) minerarie; (2) energetiche; (3) alimentari.
— *Tipo di popolazione*: primo giocatore o secondo giocatore.
— *Installazioni possedute*: (1) minerarie; (2) energetiche; (3) militari; (4) alimentari.
— *Centri di ricerca*: (1) trasporti; (2) energie; (3) alimenti; (4) medicinali.
— *Quantità dei prodotti giornalieri*: (1) minerali; (2) energie; (3) alimenti; (4) armamenti.
— *Numero degli abitanti.*
— *Tipi di energie possedute* (all'interno del pianeta).
— *Tipi di minerali posseduti* (all'interno del pianeta).
— *Tipi di alimenti posseduti* (all'interno del pianeta).
— *Tipi di armamenti costruiti.*
— *Superficie utilizzabile.*
— (qualunque altra cosa vi verrà in mente!!!... non esagerate).

Inoltre ognuna delle due civiltà possiede delle caratteristiche che sono valide per tutti i pianeti che ne fanno parte e sono:
— *tipi di energie*
— *tipi di armamenti*
— *tipi di alimenti*
— *tipi di mezzi di trasporto*
— *tipi di medicinali*

tutti i prodotti ottenuti con un buon impiego dei centri di ricerca. Quello che bisogna ancora fare (oltre ad eventuali modifiche di quanto è stato fatto finora) sono le leggi che regolano tutto il gioco e quindi la stesura di una specie di «regolamento»...

Scuola di videogame

Un digitalizzatore audio

Come preannunciato la volta scorsa, in questo numero ci dedicheremo alla costruzione di un digitalizzatore audio, quel dispositivo che permette di «far entrare» nel nostro computer suoni provenienti dal mondo esterno. Inutile dire che una volta «dentro» i suoni possono essere utilizzati come più ci piace (ad esempio, possono essere inseriti in un videogioco...).

Come potete notare dalle foto 1 e 2, il circuito è molto semplice e richiede un numero esiguo di componenti, quindi il suo montaggio non dovrebbe comportare grossi problemi. Oltre ai componenti elencati nella foto 1 sono necessari:

— un connettore simile a quello di un comune joystick;
— una manciata di fili per i collegamenti;
— una basetta «millefori» sulla quale realizzare il montaggio, oppure un circuito stampato disegnato tenendo conto dei collegamenti dello schema elettrico della foto 2;
— un saldatore per circuiti elettronici;
— dello stagno per le saldature;
— uno zoccolo (a 20 pin) per il circuito integrato.

Il succo di tutto il dispositivo chiaramente sta nel circuito integrato ADC 0802, che come suggerisce la sigla è un convertitore digitale analogico completo, ma che tuttavia necessita di alcuni componenti esterni.

La rete R-C serve a regolare la frequenza di un clock interno. Il partitore resistivo formato da R1 e R2 serve a regolare la tensione di riferimento, il che ci permette di adattare il circuito alla conversione di un segnale avente una qualunque ampiezza. R1 è il potenziometro che permette di effettuare le regolazioni in fase di «campionamento». Il segnale sonoro deve essere prelevato dall'uscita per cuffia del vostro stereo o del vostro registratore o di qualunque altra diavoleria avete a disposizione. Detto segnale va applicato al piedino 6 del circuito integrato (quello che nella foto 2 è indicato con Vin).

Il piedino 20 va collegato all'alimentazione, che nel nostro caso è prelevata direttamente dalla porta joystick del C64. I piedini dall'11 al 17 sono le uscite del segnale convertito. Ne utilizzeremo solo 4 visto che, come detto la volta scorsa, la risoluzione del convertitore (interno) digitale-analogico del nostro Commodore è di soli 4 bit. Prenderemo i 4 bit più significativi lasciando scollegati gli altri.

Una raccomandazione: non toccate i piedini dell'ADC con le dita altrimenti rischiate di distruggerlo (no, non è una bomba); si tratta di un integrato fabbricato in tecnologia CMOS in quanto tale è molto sensibile alle scariche elettrostatiche (quindi tenetelo lontano anche dal vostro monitor).

Il connettore a nove piedini va collegato come in foto 3. Fate attenzione alle saldature che effettuerete su quest'ultimo; non è difficile ritrovarsi con qualche corto circuito di troppo. L'importante è non utilizzare una quantità eccessiva di stagno. Fondamentale è l'utilizzo di uno zoccolo per il collegamento dell'IC sia perché in questo modo potrete facilmente asportarlo dal circuito, sia perché (cosa molto più importante) è indispensabile non sottoporlo al calore del saldatore.

Considerata la semplicità del circuito, la costruzione risulta agevole anche su una basetta «millefori» o simili, quindi se il vostro lavoro non deve essere presentato ad una mostra di arte elettronica è preferibile lasciar da parte acidi e trasferibili. Adottando la prima soluzione tuttavia occorre far attenzione alle solite saldature, che anche in questo caso potrebbero provocare indesiderati corto circuiti. Fate molta attenzione alla piedinatura dell'integrato; scambiare uno dei pin con un altro è uno scherzo.

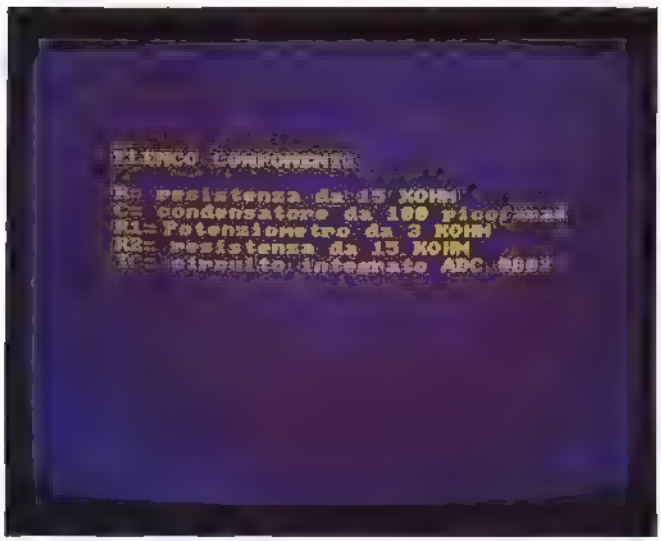


Foto 1

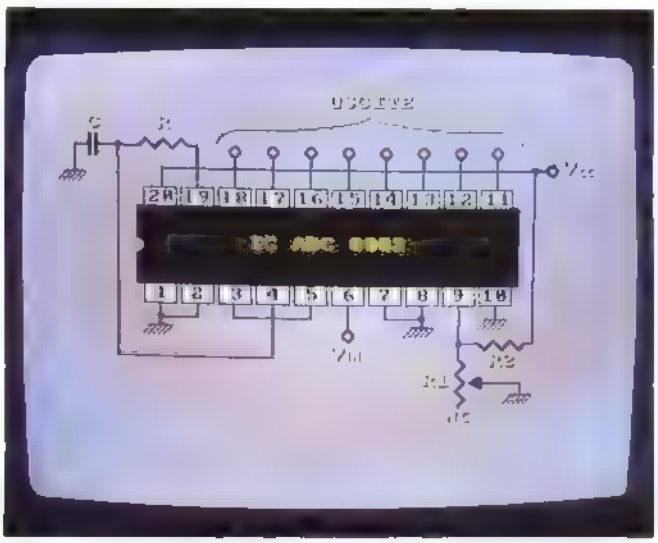


Foto 2

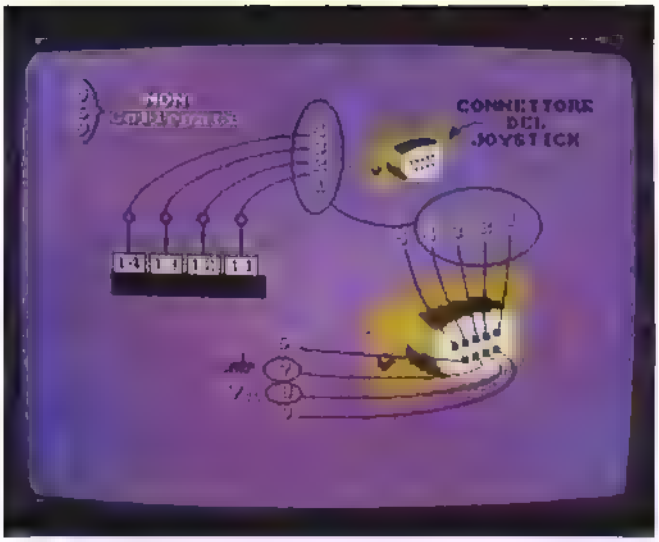


Figura 1

Foto 3

Una volta terminati tutti i collegamenti e sistemato l'integrato al suo posto (fate attenzione anche a questo) il digitalizzatore è pronto per essere usato.

A proposito di collegamenti: il segnale in ingresso da prelevare dall'uscita per cuffia del vostro apparecchio audio deve essere regolato ad un volume né troppo alto né troppo basso, ma non è una regolazione molto critica, visto che il digitalizzatore possiede di per sé un regolatore (il potenziometro).

Il programma di figura 1 è un TEST di buon funzionamento. Esso carica il valore della locazione 56320 ($DC00), che corrisponde alla porta joystick 2 e lo trasferisce nel registro 54296 ($D418) che corrisponde al «volume» del SID. Con questo programma si ha il massimo della resa del nostro apparecchio. Fatelo partire e collegate alla porta 2 il connettore del digitalizzatore. Applicate quindi un segnale all'ingresso Vin e... non aspettatevi grandi risultati. Probabilmente le prime «note» che ascolterete dall'altoparlante del vostro monitor saranno solo delle pernacchie, ma non scoraggiatevi. Regolate il potenziometro R1 e il suono si farà più nitido. Come detto questo è solo un test di buon funzionamento e quindi non permette di memorizzare nel computer il suono campionato. Ci occorre un programma in grado di farlo... e indovinate cosa ha pensato il vostro Marco Pesce? Da bravi studenti ve lo realizzerete da soli, ma niente paura, per i più pigri ne pubblicherò una mia versione sul prossimo numero. Buon lavoro!

Megaposta

Cominciamo con i ringraziamenti. Grazie a:
Ettore de Simone (Roma)
Carmine Musso (Portici, NA)
Emiliano D'onofrio (Roma)
Marco Marinai (S. Maria e Monte, PI)
Jacopo Piazzi (Siena)
Domenico Pignatelli (Potenza)
Iacopo Benevieri (Firenze)

«Caro Marco,
prima di tutto mi voglio complimentare per la rubrica che ritengo sia trattata con grande competenza. Il motivo di questa mia è il listato del programma di moltiplicazione degli sprite apparso a pagina 151 del numero di luglio/agosto.

La mia attenzione è stata attratta principalmente dalla routine di sort che si occupa di ordinare le ordinate degli sprite. Ti ho scritto perché a mio parere il tipo di algoritmo usato non è dei più veloci. A questo proposito ho scritto una routine di sort che ritengo più efficiente e che ti invio (il listato). L'algoritmo usato compie i seguenti calcoli: prima di tutto divide per due il numero di valori da ordinare e considera la tabella divisa in due tabelle distinte. Successivamente confronta il primo valore di questa tabella con il primo della seconda e se minore scambia i due valori. La stessa operazione è ripetuta per i secondi, per i terzi e via dicendo fino alla fine della tabella 1.

A questo punto le tabelle vengono a loro volta divise per due e l'operazione di confronto viene ripetuta fino a che le tabelle raggiungono la dimensione di un solo byte. Infine un flag informa la routine se nell'ultima operazione i valori erano già ordinati o meno. In caso affermativo la routine termina con un «RTS», altrimenti dà un'ultima passata ai valori fino e che non sono completamente ordinati. Colgo l'occasione per dirti che ho terminato un programma che consente di stampare schermate grafiche per i videogiochi; il concetto è simile alla gestione della grafica del «S.E.U.CK.», cioè prima si disegnano i caratteri grafici e poi si inseriscono in tessere da 5×5 caratteri e infine si disegna l'intera area grafica come se fosse un puzzle, permettendo di creare delle aree immense risparmiando notevolmente la memoria...»

Paolo Galimberti, Gorgonzola (MI)

Caro Paolo, ad essere sincero non ho avuto il tempo di visionare accuratamente la tua routine, ma dalla descrizione che mi hai fatto non mi sembra molto migliore di quella da me realizzata, soprattutto considerando che bisogna tener conto sempre del tempo massimo di ordinamento, comunque non posso pronunciarmi oltre... il tuo programma sembra interessante, inviamelo (possibilmente su dischetto) e se sarà degno di nota magari ci scappa anche un compenso!

di Marco Pesce

Un videogioco tutto nostro

undicesima parte

Come al solito lo schema delle caratteristiche di ogni singolo pianeta va nuovamente modificato (chissà perché manca sempre qualcosa). Sarà la modifica definitiva?... speriamo di sì. Avete visto quante foto? Sono tutte relative alla sezione dei controlli sui pianeti e verosimilmente saranno quelle definitive (magari con qualche modifica grafica... naturalmente in meglio). In questa nuova puntata analizzeremo i dettagli dei menu e delle interfacce messe a disposizione dai giocatori per quanto concerne l'attività strategica e inoltre quantificheremo i dati relativi ai pianeti

Le modifiche allo schema delle caratteristiche vanno fatte in «TIPO... (3) adatto solo per l'installazione di basi militari» che diventa «... (3) adatto solo come deposito», in «QUANTITÀ DI RISORSE POSSEDUTE...» dove si deve aggiungere «... (4) militari», in «CENTRI DI RICERCA...» dove si aggiunge «... (5) armamenti, (6) minerali», in «TIPI DI ARMAMENTI COSTRUITI» che non serve più e va cancellato, in «TIPI DI ARMAMENTI» dove si sostituisce «POTENZA DEGLI ARMAMENTI» e infine si deve aggiungere «TIPI DI MINERALI».

La vita di un pianeta

Consideriamo un pianeta abitabile. La quantità di risorse possedute viene continuamente alterata con il passare del tempo. Ad esempio quelle alimentari dipendono dal numero di abitanti, dalle installazioni alimentari possedute e dai tipi di alimenti che il pianeta possiede in natura e che quindi possono essere prodotti con le installazioni a parità di costi di produzione. Chiaramente ogni pianeta possiede caratteristiche naturali diverse e quindi ci saranno pianeti che ne hanno molti (e quindi vanno sfruttati maggiormente) ed altri che ne hanno pochi. Il numero di tipi di alimenti tuttavia potrebbe essere molto più elevato anche su pianeti apparentemente non adatti ad uno sfruttamento alimentare; tutto dipende dai centri di ricerca che quando scoprono un nuovo alimento questo viene riscontrato producibile solo in una parte dei pianeti che compongono la galassia. Una volta calcolata la produzione giornaliera di un pianeta essa andrà a sommare alla quantità di risorse disponibili (quindi già prodotte), ma da questa quantità deve essere sottratto il fabbisogno della popolazione.

Anche per le altre risorse il discorso è molto simile, più è alto il numero di tipi più elevata sarà la produzione.

I centri di ricerca incrementano la loro efficienza con l'incrementarsi della loro quantità; in altre parole, più ce ne sono, più probabilità abbiamo di fare nuove scoperte.

Vediamo ora di quantificare un po' di dati. La superficie di un pianeta (in kmq) va da un minimo di 200.000.000 a un massimo di 1.000.000.000. Considerando che il 70% della superficie è occupato dalle acque, la superficie utile va da un minimo di 60.000.000 a un massimo di 300.000.000. Nei dati verrà espressa solo la superficie utile. In un kmq entrano al massimo 100 abitanti quindi nel pianeta più piccolo il massimo virtuale sarà di 6.000.000.000 di abitanti. Virtuale perché chiaramente nel pianeta ci sono anche le installazioni che occupano spazio (1 kmq per installazione, qualunque essa sia).

I rendimenti per ogni singola installazione corrispondono a:
Mineraria: 100 unità
Energetica: 100 unità
Alimentare: 1000 unità
Militare: 10 unità
mentre le richieste sono:
Mineraria: 10 unità di energia, 0 unità di minerali
Energetica: 0 unità di energia, 0 unità di minerali
Alimentare: 100 unità di energia, 0 di minerali
Militare: 100 di enegia, 1000 di minerali.

Per i centri di ricerca i costi di mantenimento sono diversi:
Trasporti: 10 di energia, 100 di minerali
Energia: 20 di energia, 10 di minerali
Alimenti: 5 di energia, 5 di minerali
Medicinali: 10 di energia, 10 di minerali
Armamenti: 100 di energia, 200 di minerali
Minerali: 10 di energia, 0 di minerali

Le possibili scoperte invece sono rispettivamente: mezzi più veloci e/o meno costosi, nuove energie, nuovi alimenti, medicinali più efficaci, armamenti più potenti, nuovi minerali.

I consumi degli abitanti (per 100 abitanti):
Alimenti: 100 unità
Energie: 1 unità

A differenza di quanto succede con gli altri centri di produzione, il numero di armamenti costruiti resta stabile e non aumenta in funzione delle scoperte dei centri di ricerca, aumenta invece la loro potenza.

Quando una delle risorse non è più sufficiente si entra in stato di allarme e cioè il computer avverte il giocatore della particolare condizione critica. Se la situazione non viene ristabilita cominciano i guai, ovvero cominciano a bloccarsi alcuni centri oppure muoiono alcune persone. Quando ci accorge che una

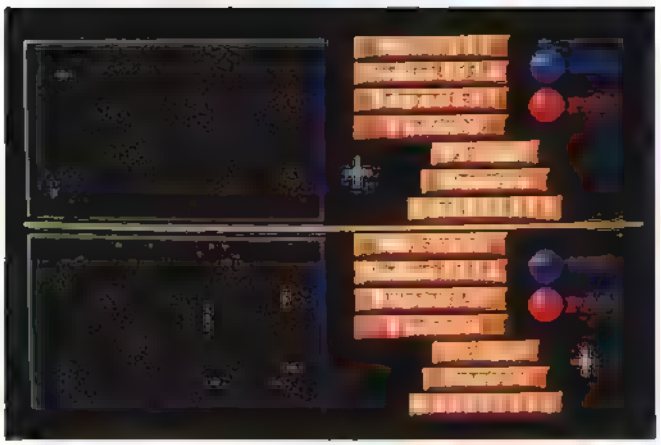


Foto 1

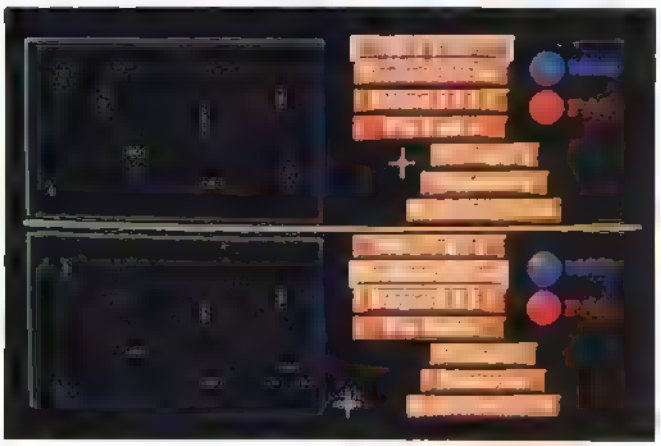


Foto 2

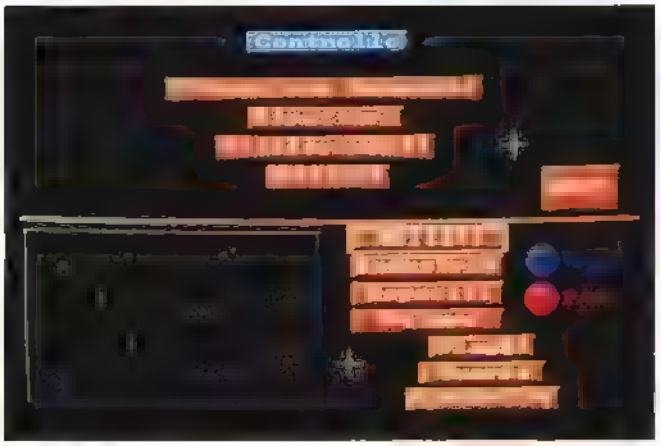


Foto 3

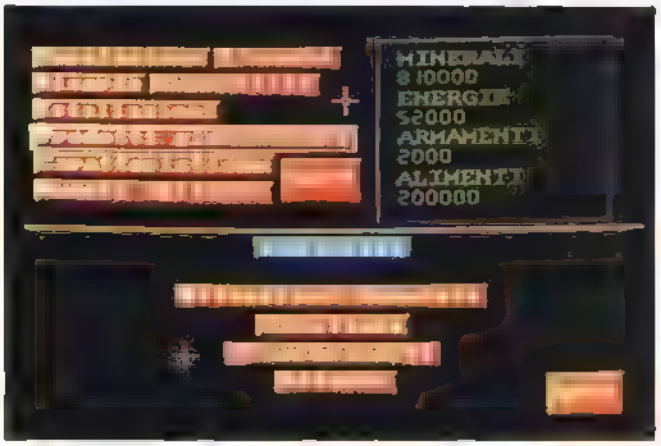


Foto 4

certa risorsa comincia a scarseggiare è opportuno disattivare parte delle installazioni che non producono materiale di elevata necessità in modo da eliminare il consumo della installazione stessa.

Osserviamo ora i costi di costruzione per i centri produttivi:

Installazione energetica: 10.000 unità di energia, 1000 unità di minerali

Install. mineraria: 1000 di energia, 1000 di minerali

Install. alimentare: 1000 di energia, 1000 di minerali

Install. militare: 10.000 di energia, 10.000 di minerali

e per quelli di ricerca:

Trasporti: 1000 di energia, 10.000 di minerali

Energie: 1000 di energia, 1000 di minerali

Alimenti: 1000 di energia, 1000 di minerali

Medicinali: 1000 di energia, 1000 di minerali

Armamenti: 10.000 di energia, 1000 di minerali

Minerali: 1000 di energia, 1000 di minerali.

I costi dei viaggi di trasporto materiali dipendono dalla distanza da percorrere e dal carico da trasportare.

Gli abitanti del pianeta tendenzialmente crescono di numero ma chiaramente ci sono anche le morti; tuttavia queste possono essere attenuate dalla scoperta di medicinali più potenti.

I menu

Il programma dovrà gestire: la vita di tutti i pianeti, i viaggi, le battaglie, le sezioni ARCADE, il controllo dei pianeti, il controllo delle guerre, il controllo dei viaggi, gli imprevisti.

La foto 1 mostra il primo schermo (di gioco vero e proprio) che vedranno i due giocatori. Sulla sinistra è visualizzata la mappa della galassia (una microgalassia) per intero, che comprende 59 pianeti fra i quali 20 sono abitati, 18 contengono solo installazioni (industrie) e 21 vengono usati come depositi. I sistemi solari (o stellari... come volete) sono 10. Sulla destra troviamo invece una prima serie di comandi; i due Led giganti simboleggiano il primo e il secondo giocatore e ad ognuno di essi è associata una sigla inserita preventivamente dai giocatori stessi. Ogni giocatore ha a sua disposizione un cursore-puntatore e una sezione dello schermo. Il puntatore, ovviamente manovrabile tramite joystick, ci permette di agire sui comandi. Alcuni comandi servono per alterare la visuale della mappa. Questi sono i due led e i tre «pulsanti» contrassegnati con «abitati», «industrie» e «depositi». Posizionando il puntatore su uno di essi e pigiando sul tasto di fuoco del joystick, si attiva il comando. Attivando uno dei due led si elimina dalla mappa la visualizzazione dei pianeti appartenenti al giocatore a cui si riferisce il led non attivato. Attivando ancora uno dei tre pulsanti, ad esempio quello contrassegnato con «abitanti» (foto 2) si eliminano dalla mappa anche tutti i pianeti che non sono abitati (afferrato il

Foto 5

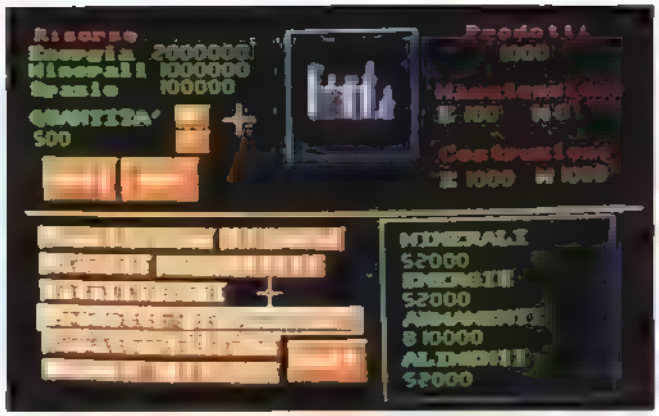


Foto 6

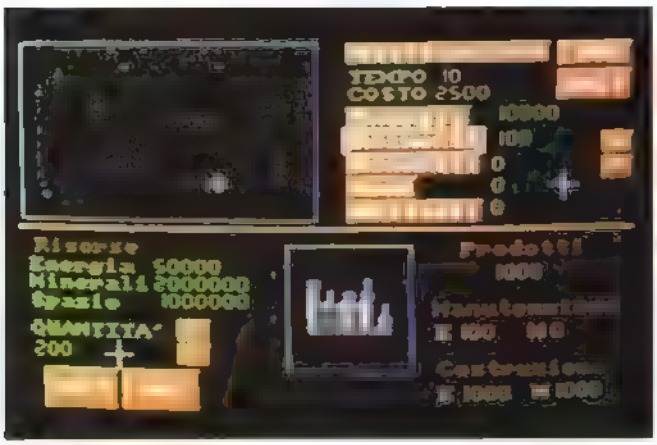


Foto 7

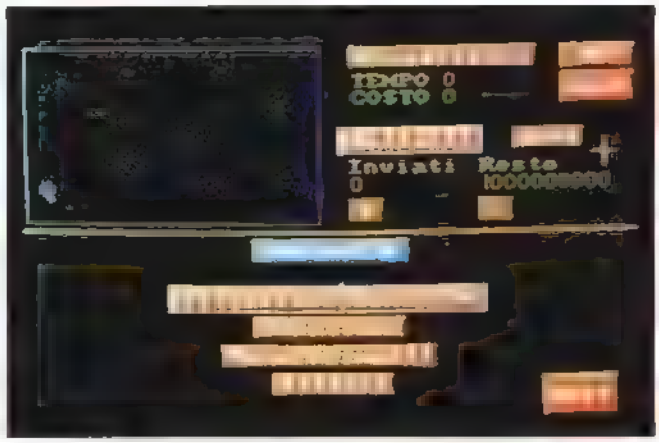


Foto 8

meccanismo?) Chiaramente per disattivare un comando basterà «ripigiarlo».

Esaminiamo gli altri comandi. Il comando controllo ci abilita a selezionare uno dei pianeti correntemente visualizzati sulla mappa.

Una volta selezionato (sulla mappa appare anche un comando fantasma di QUIT per i casi di errore) si entra nel modo «controllo».

Controllo di un pianeta

A questo punto viene visualizzato un nuovo panello (quello della foto 3) che ci permette di:

— osservare la situazione generale di un pianeta;

— costruire una nuova struttura o disattivarne alcune o riattivarle o distruggerle (solo se è uno dei nostri pianeti),

— trasferire i prodotti di un pianeta su un altro (solo se sono entrambi in nostro possesso);

— attaccare il pianeta (solo se è in possesso dell'avversario).

Se viene selezionato il comando di controllo della situazione generale del pianeta ci troveremo di fronte allo screen rappresentato nella foto 4. Ognuno dei pulsanti posti sulla sinistra, se attivato, fara visualizzare nel sub-screen a destra le informazioni relative a quel determinato campo specifico (tipo di pianeta, tipo di popolazione, quantità delle risorse, delle installazioni, dei centri di ricerca, etc.). Azionando il pulsante di QUIT chiaramente si tornerà allo screen di foto 3.

Se viene azionato il comando relativo alle strutture passa a uno screen che ci permette di decidere se vogliamo costruire, disattivare, riattivare o distruggere e poi allo screen di foto 5.

Con questo nuovo pannello possiamo scegliere il tipo di costruzione che nel sub-screen centrale sarà visualizzata graficamente (anche per questioni di look). Sulla destra troviamo invece un altro pannello che ci informa sulle caratteristiche della costruzione.

Possiamo, una volta definita la costruzione da trattare, passare ad un sub-menu (con l'apposito tasto) che fa visualizzare lo schermo di foto con il quale il giocatore può selezionare la quantità. Se ad esempio decidiamo di costruire un centro di produzione alimentare, una volta giunti a questo sub-menu dovremo selezionare il numero di questi centri e verremo informati sulla restante quantità di energia, di minerali e di spazio. Dato l'OK si ritorna allo schermo di foto 5 e se non dobbiamo costruire altro possiamo selezionare il QUIT. È da notare che su un pianeta NON ABITABILE si possono costruire solo installazioni minerarie energetiche o militari oppure non si può costruire niente nel caso di pianeti-deposito. Ritornamo al menu di controllo; se si seleziona l'opzione di trasferimento dei prodotti è lo schermo di foto 7 ad essere visualizzato. Abbiamo nuovamente a disposizione la mappa, dalla quale selezioneremo il pianeta destinazione (con l'apposito comando), più dei comandi che ci permettono di determinare la quantità di materiali da trasferire. Ovviamente sulla mappa sono presenti esclusivamente i pianeti in nostro possesso. Quando si seleziona il pianeta destinazione viene visualizzato per esso il pannello delle caratteristiche generali. È evidente che gli abitanti non possono essere trasferiti su un pianeta non abitabile. Naturalmente sul

pianeta che offre viene reso disponibile nuovo spazio, mentre viene sottratto a quello che riceve.

Lo spazio occupato dai materiali è il seguente:

1000 unità di energia: 1 kmq
1000 di minerali: 1 kmq
1000 di alimenti: 1 kmq
1000 di armamenti: 10 kmq.

In due display vengono visualizzati i costi e i tempi necessari per il trasferimento. Anche qui possibilità di OK ■ di QUIT.

L'ultima possibilità del menu dei controlli è quella relativa all'ATTACCO del pianeta. La foto ■ riguarda il pannello di questo comando che è simile a quello necessario per i trasferimenti. La mappa anche in questa circostanza presenterà esclusivamente i pianeti in nostro possesso, dai quali, appunto, verrà dichiarata guerra. Ad ogni abitante corrisponde una sola macchina da guerra.

Anche qui ci sarà il preventivo dei costi e dei tempi di arrivo della «flotta». Una volta dato l'OK e una volta arrivata la flotta, la guerra ha inizio e verrà gestita dal computer a meno che il giocatore non vorrà parteciparvi di persona; in tal caso sarà messo al comando di una delle navi spaziali.

Per questa puntata è tutto, nella prossima esamineremo i menu restanti e le altre «cosucce» e cominceremo a fare un prospetto delle routine che ci occorrono.

Scuola di videogame

Un digitalizzatore audio (il software)

Siete riusciti nell'impresa? Non ditemi che non avete neppure cominciato... non sapete cosa vi perdete. E il programma?... Eccolo qui!

Passiamo subito al sodo: il programma che pubblichiamo in questa pagina permette di memorizzare in RAM il segnale audio-digitale proveniente dalla porta 2 e di riascoltarlo quante volte vogliamo. La prima routine che parte con

SYS 40448 ($9E00)

è quella di PLAY ovvero quella che riproduce il suono già memorizzato in RAM ■ che può essere inserita nei vostri elaborati, unitamente ai dati digitali, per «effetti parlati» (o anche cantati...). L'altra è quella di RECORD che naturalmente va usata insieme al digitalizzatore per... «campionare» ■ si attiva con

SYS 40607 ($9E90)

Essa comprende una routine di «attesa» che ci permette di ascoltare il segnale proveniente dall'esterno e di attivare al momento opportuno la «registrazione» con la pressione di un qualunque tasto.

Durante il campionamento non si ascolta nessun suono. È da notare il fatto che il digitalizzatore non deve essere lasciato inserito nella porta joystick quando non lo usiamo, altrimenti la scrittura da tastiera viene alterata (ovvero se ■ premono dei tasti compaiono lettere che non c'entrano niente). La procedura completa da seguire per digitalizzare un suono è la seguente:

— impostate la SYS 40607;
— inserite il digitalizzatore nella porta 2;
— attivate il dispositivo generatore di suono collegato all'ingresso del digitalizzatore;
— ■ questo punto si dovrà udire il suono dall'altoparlante;
— regolate il suono con il potenziometro del digitalizzatore;
— quando siete pronti (ovvero quando volete digitalizzare il suono) premete un tasto della tastiera del C-64;
— il computer si ammutolisce per qualche secondo;
— quando riappare il cursore lampeggiante la digitalizzazione è completata e potete quindi scollegare il digitalizzatore.

Da questo momento il suono è entrato nel computer e potete riascoltarlo impostando la SYS della routine di PLAY.

La durata della digitalizzazione può essere variata «pokando» nella locazione 40513 un valore compreso tra 1 e 255. Più è elevato tale valore più lungo sarà il tempo di digitalizzazione, ma naturalmente peggiore sarà il risultato in termini di qualità sonora. Questo perché la «pokata» va ad alterare la frequenza di campionamento. A tal proposito è opportuno far notare che un range accettabile di possibili valori da inserire nella sopracitata locazione è quello che va da 1 a 20.

Fin qui le istruzioni per l'utente; forniamo ora qualche informazione ■ chi vuole «capire» il funzionamento del programma.

Il programma di RECORD deve memorizzare in RAM tutti i valori provenienti, durante il campionamento, dai 4 bit meno significativi della locazione 56320 ($DC00), che equivalgono allo stato del digitalizzatore. Essendo un byte composto da 8 bit sarebbe uno spreco occupare un byte per ogni stato, quindi in un byte memorizzeremo due stati (uno nei 4 bit meno significativi e

```
   PC  IRQ  SR AC XR YR BP
;E378 EA31 FF FF 00 00 FF
.9E00 78        SEI
.9E01 A0 00     LDY #$00
.9E03 A9 00     LDA #$00
.9E05 85 FB     STA $FB
.9E07 A9 05     LDA #$05
.9E09 85 FC     STA $FC
.9E0B B1 FB     LDA ($FB),Y
.9E0D 29 0F     AND #$0F
.9E0F 8D 1B D4  STA $D41B
.9E12 20 40 9E  JSR $9E40
.9E15 B1 FB     LDA ($FB),Y
.9E17 4A        LSR
.9E18 4A        LSR
.9E19 4A        LSR
.9E1A 4A        LSR
.9E1B EA        NOP
.9E1C EA        NOP
.9E1D EA        NOP
.9E1E 8D 1B D4  STA $D41B
.9E21 20 40 9E  JSR $9E40
.9E24 C8        INY
.9E25 F0 03     BEQ $9E2A
.9E27 4C 0B 9E  JMP $9E0B
.9E2A E6 FC     INC $FC
.9E2C A5 FC     LDA $FC
.9E2E C9 BF     CMP #$BF
.9E30 F0 03     BEQ $9E35
.9E32 4C 0B 9E  JMP $9E0B
.9E35 58        CLI
.9E36 60        RTS
.9E37 EA        NOP
.9E38 EA        NOP
.9E39 EA        NOP
.9E3A EA        NOP
.9E3B EA        NOP
.9E3C EA        NOP
.9E3D EA        NOP
.9E3E EA        NOP
.9E3F EA        NOP

.9E40 A2 01     LDX #$01
.9E42 CA        DEX
.9E43 D0 FD     BNE $9E42
.9E45 60        RTS
.9E46 EA        NOP
.9E47 EA        NOP
.9E48 EA        NOP
.9E49 EA        NOP
.9E4A EA        NOP
.9E4B EA        NOP
.9E4C EA        NOP
.9E4D EA        NOP
.9E4E EA        NOP
.9E4F EA        NOP
.9E50 78        SEI
.9E51 A0 00     LDY #$00
.9E53 A9 00     LDA #$00
.9E55 85 FB     STA $FB
.9E57 A9 09     LDA #$09
.9E59 85 FC     STA $FC
.9E5B AD 00 DC  LDA $DC00
.9E5E 29 0F     AND #$0F
.9E60 91 FB     STA ($FB),Y
.9E62 20 40 9E  JSR $9E40
.9E65 AD 00 DC  LDA $DC00
.9E68 0A        ASL
.9E69 0A        ASL
.9E6A 0A        ASL
.9E6B 0A        ASL
.9E6C 11 FB     ORA ($FB),Y
.9E6E 91 FB     STA ($FB),Y
.9E70 20 40 9E  JSR $9E40
.9E73 C8        INY
.9E74 F0 03     BEQ $9E79
.9E76 4C 5B 9E  JMP $9E5B
.9E79 E6 FC     INC $FC
.9E7B A5 FC     LDA $FC
.9E7D C9 BF     CMP #$BF
.9E7F F0 03     BEQ $9E84

.9E81 4C 5B 9E  JMP $9E5B
.9E84 58        CLI
.9E85 60        RTS
.9E86 EA        NOP
.9E87 EA        NOP
.9E88 EA        NOP
.9E89 EA        NOP
.9E8A EA        NOP
.9E8B EA        NOP
.9E8C EA        NOP
.9E8D EA        NOP
.9E8E EA        NOP
.9E8F EA        NOP
.9E90 AD 00 DC  LDA $DC00
.9E93 8D 1B D4  STA $D41B
.9E96 A5 C6     LDA $C6
.9E98 C9 00     CMP #$00
.9E9A D0 B4     BNE $9E50
.9E9C 4C 90 9E  JMP $9E90
.9E9F A0 00     LDY #$00
.9EA1 A9 00     LDA #$00
.9EA3 85 FB     STA $FB
.9EA5 A9 09     LDA #$09
.9EA7 85 FC     STA $FC
.9EA9 A9 00     LDA #$00
.9EAB 91 FB     STA ($FB),Y
.9EAD C8        INY
.9EAE D0 FB     BNE $9EAB
.9EB0 E6 FC     INC $FC
.9EB2 A5 FC     LDA $FC
.9EB4 C9 BF     CMP #$BF
.9EB6 D0 F1     BNE $9EA9
.9EB8 EA        NOP
.9EB9 EA        NOP
.9EBA EA        NOP
.9EBB EA        NOP
.9EBC EA        NOP
.9EBD EA        NOP
.9EBE EA        NOP
.9EBF EA        NOP
```

Listato del digitalizzatore

uno nei 4 bit più significativi). Per poter selezionare di volta in volta una locazione differente, il programma usa il metodo detto INDIRETTO INDICIZZATO che in questo caso viene usato nell'istruzione

STA ($FB),Y

e che permette di «puntare» ad una qualunque locazione RAM semplicemente alterando il valore contenuto in una locazione di memoria e nel registro Y. In altre parole la succitata STA deposita il valore contenuto nell'accumulatore nella locazione di memoria che si ricava utilizzando come byte alto il contenuto della locazione $FC e come byte basso il contenuto della locazione $FB e addizionando al risultato di questo abbinamento il valore contenuto nel registro Y.

Ad esempio (ci vuole proprio...) se nella locazione $FC depositiamo il valore #$09 e nella $FB il valore #$00 e nel registro Y il valore #$00, l'istruzione

STA ($FB),Y

deposita il contenuto dell'accumulatore nella locazione $0900.

A prima vista questo sistema sembra solo una complicazione di quello visto qualche puntata fa e cioè

STA $0900,Y

ma in realtà quest'ultimo non permetteva di accedere a tutte le locazioni della RAM, ma al massimo a 256 di queste, come noto, alterando il contenuto del registro Y. Con questo nuovo metodo invece basta cambiare il contenuto delle locazioni $FB e $FC e possiamo «puntare» a tutta la RAM. Il metodo di indirizzamento appena visto funziona solo in «pagina zero» ovvero con le prime 256 locazioni di RAM, quindi al posto di $FB è possibile sostituire ad esempio $FD ma non $0100.

Nel programma vengono inizializzate le locazioni $FB e $FC proprio con i valori #$09 e #$00, ma è possibile cambiarli a piacimento (in tal modo si altera la locazione di partenza per la memorizzazione dei dati digitali). La prima istruzione «attiva» del programma è la LDA $DC00. Segue la AND #$0F che elimina i 4 bit meno significativi; il risultato viene memorizzato nei 4 bit meno significativi della locazione $0900. Si esegue ora una routine di pausa che serve per rallentare la frequenza di campionamento. Si ricarica la locazione $DC00 nell'accumulatore (che probabilmente ora conterrà un nuovo valore) e il suo contenuto viene traslato di quattro posizioni a sinistra nel byte «ORAto» con quello della locazione $0900 e infine memorizzato in $0900. Tutta questa procedura serve per memorizzare nei 4 bit più significativi i 4 bit meno significativi della locazione $DC00, che andranno quindi ad occupare lo spazio avanzato in precedenza nella locazione $0900. Notiamo a questo punto che se nei quattro bit più significativi erano già presenti degli UNO succedeva un disastro in quanto l'operazione di or logico nel nostro contesto presuppone che tali 4 bit siano tutti a zero. Proprio per evitare il succitato inconveniente, prima di iniziare con la digitalizzazione vera e propria la parte di ram che ospiterà il segnale digitalizzato viene ripulita da un'apposita routine. Tuttavia l'inconveniente si poteva evitare in altro modo, azzerando tali bit di volta in volta che venivano usati (con una operazione di and logico simile a quella vista in precedenza) ma ciò avrebbe comportato un ulteriore spreco di tempo, infatti già la ORA($FB),Y varia i tempi di esecuzione di questa routine rispetto alla routine di PLAY (dove tale istruzione è stata sostituita da 3 NOP proprio per «bilanciare» i tempi, cosa che se non effettuata produce una differenza tra velocità di registrazione e velocità di riproduzione).

Il resto della routine non è degno di particolare attenzione e anche la routine di PLAY presenta istruzioni già viste, quindi possiamo anche salutarci. Ciao!

Megaposta

Le scuse del... «filosofo»

«Caro ingegnere Pesce (spada) Marco,
devo assolutamente porgerle le mie più umili e latifondiste scuse, per il fatto che vi avevo accusato ignobilmente di creare un palloso clone di ELITE e compagnia bella, ma mi sono accorto che in realtà il vostro (anzi il nostro) giochettuolo è talmente ampio da soddisfare qualsiasi videogiocatore. Spero solo che venga realizzato con cura, perché sulla carta, il Megagame, ha tutte le carte in regola (scusate il gioco di parole) per diventare un PIG, ehm volevo dire BIG.

La saluto e offro il mio aiuto, perché, pur non essendo valido dal punto di vista della programmazione, sono un criticone inguaribile, e non sempre dalla penna avvelenata.

Distinti saluti».

Matteo Bittanti,
S. Pietro All'Olmo (MI)

Mi spiace ma non posso accettare le tue scuse (... scherzo).

Arriva la «replica degli altri»

«— Staremo a sentire la replica degli... altri —. Così concludesti, caro Marco, il commento sulla lettera di Paolo Costabel apparsa su MC n. 77.

Ed è proprio una replica che voglio fare.

Chi va cianciando sulla «simulazione totale» e si augura l'avvento di computer sempre più potenti HA CAPITO che il futuro di qualsiasi programma, sia «giocoso» che «serioso» dipende dalla potenza stessa. Pochi anni or sono era impensabile poter presentare documenti con grafici non solo a torta o a barre. La scelta era obbligata. Oggi possiamo ottenere una vasta tipologia di grafici (vedi pag. 67 di MC n. 76).

Dirò di più! Sono stati i «simulatori» a spingere i programmatori nell'ottenere immagini sempre migliori. Prendi ad es. il primo POLE POSITION per 64 e TEST DRIVE!!! E se pensi che la computer simulazione sia un rimbecillitore di masse... allora non sei mai stato nei simulatori dell'Alitalia. Se non puoi venire a Fiumicino... leggiti la recensione di Giustozzi su MC n. 65!

E quando giocherai con ranocchie e biglie, e ne sarai entusiasta... ricordati di chi ha «creato» quelle tecniche...

P.S. Complimenti per la rubrica».

Mauro Luis Longarini, Ostia (ROMA)

Caspita se la sono presa davvero! Caro Mauro, probabilmente hai frainteso quanto Paolo voleva esprimere con la sua lettera. Nessuno pensa che la computer simulazione sia un «rimbecillitore di masse», nessuno vuole fermare l'evoluzione dei computer, ma qui si parla di videogiochi!!! Credi veramente che la bellezza di un videogame sia tutta nel vedere sullo screen, riprodotti in modo stupendo, 4 modellini di automobili? Il problema che riproponeva Paolo era quello relativo alla mancanza di fantasia da parte dei programmatori. Non basta avere un computer super potente per creare un gioco nuovo.

Un altro programmatore

«Ciao Marco...
essendo un lettore della rivista MC ed essendo particolarmente interessato all'articolo sul Megagame 64, ti ho scritto per dirti che se hai bisogno di programmatori in L.M....».

Stefano Degan, Legnago (VR)

Dovrai aspettare la prossima puntata... (nel frattempo non cambiare idea mi raccomando!).

di Marco Pesce

Un videogioco tutto nostro

Siamo ormai alle battute finali nella definizione del nostro megagame. Tra poco è Natale, e non possiamo ancora mettere sotto l'albero il super regalone che pian piano stiamo costruendo. Non arriveremo al prossimo dicembre, vero?

Cosa manca da definire? Cominciamo a farci qualche domanda

1) Che succede quando un pianeta viene conquistato?
2) Che fine fanno le astronavi che si ritirano o che vincono, ma non possono restare sul pianeta perché non abitabile?
3) Si possono inviare rinforzi durante una battaglia?
4) Come si entra nel modo soccorsi (foto 1)?
5) Come si partecipa attivamente a una battaglia?

Ecco le risposte:
1) Il pianeta diventa di «proprietà» del vincitore. A proposito: le due civiltà rivali sono strutturate in maniera totalmente diversa e le tecnologie di una non vanno bene per l'altra. Conseguenza: ogni volta che conquistiamo un pianeta dobbiamo «ristrutturarlo» da zero, quindi con nuove costruzioni, nuove risorse di base etc. Se vi state chiedendo quale sia il destino dei vecchi abitanti non avete afferrato il concetto: «non sono dei nostri... tutti morti!».
2) Non preoccupatevi: o restano (se il pianeta è abitabile) oppure tornano indietro. Quest'ultima situazione si traduce in un viaggio che viene gestito dall'apposita routine addetta.
3) La risposta ovviamente è sì. Bisognerà commissionare un trasporto di astronavi sul pianeta designato con l'opzione «trasferimenti». Questo vale sia per gli attaccanti che per i difensori.
4) Ci si entra automaticamente non appena si verifica una delle situazioni seguenti:
— c'è una carenza di risorse
— c'è un eccesso di risorse (finito lo spazio a disposizione!)
— siamo attaccati
— si verifica un imprevisto (poi vedremo quali possono essere);
chiaramente quello che dovremo fare è rimediare alla situazione di disagio. Quando uno dei nostri pianeti sta per essere attaccato da una flotta nemica (ovvero non appena il giocatore avversario fa partire la spedizione!), si entra nel modo soccorso esclusivamente per avvertire il giocatore. Eventualmente questi deciderà se è il caso di inviare rinforzi. La battaglia in ogni caso sarà gestita dall'opzione dedicata.
5) Una battaglia ha inizio automaticamente non appena un pianeta viene raggiunto dalla flotta nemica. Gli abitanti del luogo risponderanno al fuoco senza richiedere l'intervento del giocatore. Ma come si fa dunque a influenzare l'esito del conflitto? È possibile partecipare all'azione teleguidando, dalla nostra postazione, una delle navicelle inviate. Questa possibilità ovviamente si può sfruttare selezionando l'opzione relativa alle guerre.

A questo punto mi viene in mente

Foto 1

un'altra domanda: che succede se ci rifiutiamo di soccorrere un pianeta, o se i soccorsi non sono sufficienti o, ancora, che succede durante il tempo di transizione fra la «richiesta» e la «risposta»? Nel primo ■ nel secondo caso ■ subiscono i danni e dopo non molto ci sarà una nuova richiesta di soccorso. Nel terzo caso ■ subiscono i danni fin quando i soccorsi non arrivano.

Da notare che non esiste la possibilità di rivolta della popolazione (ricordiamoci che le popolazioni sono molto evolute ed hanno moltissima fiducia nel loro «imperatore»... anche se «apparentemente» egli sta sbagliando).

La gestione delle guerre

Un pannello di comando (foto 2) ci permette di selezionare la visione della situazione di una delle battaglie in corso; situazione in termini di armamenti ■ nostra disposizione ■ ■ disposizione del nemico ■ in termini di potenza effettiva della nostra flotta ■ di quella avversaria. Per ogni battaglia possiamo decidere se è il caso di ritirarci o meno, ma ciò è fattibile solo se siamo la popolazione attaccante (si sa, un pianeta non ■ cede fino all'ultimo istante...). Da questo menu possiamo passare direttamente ■ quello relativo ai trasporti. Ciò ci darà la possibilità di inviare rinforzi. È inoltre possibile osservare le condizioni generali del pianeta (che attacchiamo ■ difendiamo); l'eventuale selezione di questa opzione ci mostrerà un pannello simile a quello della foto 4 del numero di novembre.

Il controllo dei viaggi

Esistono due tipi di viaggi:
1) trasporto di materiali vari
2) viaggio di forze armate per attacco, difesa, o ritorno.

Ognuno di essi viene controllato da punto di vista del tempo che rimane da trascorrere affinché venga raggiunta la destinazione. Un pannello di controllo qwerty qwerty... accidenti mi si sono mescolate le dita... dunque dicevo un pannello di controllo (foto 3) permette di selezionare la singola visualizzazione di un particolare viaggio. Tutto qui.

Gli imprevisti

Elenchiamo ora una serie di imprevisti che possono capitare ad ognuno dei due giocatori:
— daurante un viaggio il carico va disperso a causa di una tempesta spaziale o per l'improvvisa rottura dei motori;
— un incredibile cataclisma sconvolge uno dei pianeti e lo danneggia fortemente;
— il sistema di comunicazione con l'esterno si disattiva improvvisamente e siamo costretti ■ restare in attesa finché tutto non ritorna in condizioni di normalità;
— si cercano gufi!

Quando accade uno di questi eventi appare un pannello per avvertirci (foto 4 e 5).

Potenza della civiltà

Questa è l'opzione che visualizza un prospetto (foto 6) di quelle che sono le caratteristiche globali di tutta la civiltà relativa ■ ciascun giocatore. Prospetto che comprende:
— la quantità di tipi di energie
— la potenza degli armamenti
— la quantità di tipi di alimenti
— la quantità di tipi di minerali
— la velocità ■ l'economia dei mezzi di trasporto
— la potenza dei medicinali.

A questo punto abbiamo osservato praticamente ogni aspetto del gioco, fatta eccezione per le parti arcade e per gli elementi secondari (presentazione, stile della grafica ecc.). È arrivato il momento di strutturare e definire le routine necessarie. L'osservazione principale che bisogna fare è quella relativa al fatto che praticamente ci dovrebbero essere due programmi che girano in multitasking. Questo chiaramente è il primo problema da superare. In pratica occorre duplicare tutte le routine, tranne quelle che richiedono una semplice chiamata con passaggio di parametri (e restituzione di parametri). Il multitasking non è necessario per quanto riguarda le due sezioni strategiche, che possono essere eseguite «una dietro l'altra». È necessario invece per far girare contemporaneamente una sezione arcade e una strategica e vi spiego subito perché: la sezione arcade richiede una esecuzione sincronizzata con il raster, per rendere fluida l'azione, e quindi non può attendere che il processore finisca di eseguire un loop di calcoli. Di conseguenza è opportuno farla girare in IRQ, o meglio «farle» girare in IRQ, visto che potrebbe anche verificarsi l'eventualità

Foto 2

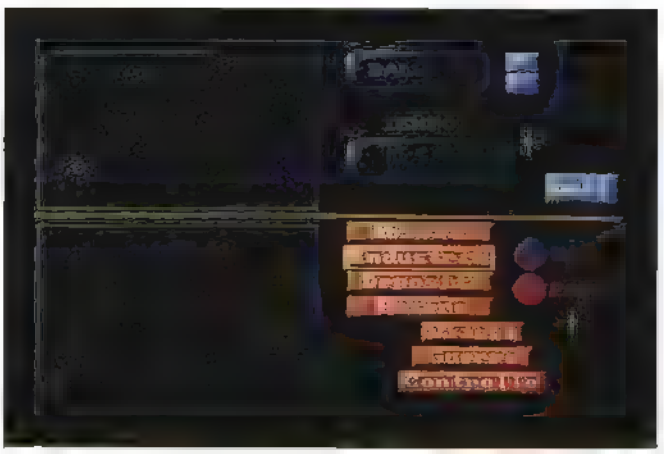


Foto 3

che i due giocatori siano contemporaneamente impegnati in una sezione arcade. Ricapitolando, le due sezioni strategiche vengono eseguite una dietro l'altra, mentre le due sezioni arcade sono eseguite in IRQ-RASTER tuttavia sempre una dietro l'altra. La sezione sonora deve anch'essa girare in IRQ e quindi seguirà a ruota dopo le sezioni arcade. L'utilizzo delle interrupt è però necessario anche per eventuali dissolvenze del quadro ■ per il movimento del cursore ecc. ■ quindi bisognerà trovare il posto anche per queste «cosucce». Certo che ad una prima occhiata la mole di lavoro che ci aspetta è veramente enorme!... e forse anche ad una seconda occhiata ■ così via...

Foto 4

Foto 5

Foto 6

Esaminiamo la routine che si occuperà della gestione della vita dei pianeti. In pratica essa contiene diverse sottoroutine adibite al calcolo delle varie risorse, tenendo conto dei numerosi fattori.

Come prima cosa dobbiamo definire l'ordine di grandezza delle varie informazioni che una struttura dati relativa a un pianeta contiene. Partiamo dalla superficie utilizzabile. Questa grandezza, come affermato la volta scorsa, oscilla tra un minimo di 60 milioni e un massimo di 300 milioni (in kmq). Per evitare una complicazione inutile dei calcoli ci conviene ridurre il massimo valore ■ 255 milioni (... che numero eh!?). Chiaramente ciò ci permetterà di memorizzare in un solo byte la «quantità di superficie» con una risoluzione di 1 milione di kmq. Qualunque costruzione occupa 1 kmq, quindi anche l'informazione relativa alla «quantità di costruzioni» esistenti può essere contenuta in un byte e con la stessa risoluzione (da notare che essendo 10 i tipi di costruzioni dovremo dedicare 10 byte ■ questa informazione). Gli abitanti in un kmq corrispondono a 100 e in questo caso, mantenendo sempre l'informazione in un solo byte, la risoluzione diminuisce ■ scende a 100 milioni di abitanti. Un discorso ■ parte si deve fare per quanto riguarda la memorizzazione della quantità di risorse a disposizione. Le risorse minerarie, così come quelle energetiche, devono necessariamente avere una risoluzione di 5 milioni di unità, in quanto, dando uno sguardo ai consumi dei centri, ci accorgiamo che un centro di ricerca alimentare consuma 5 unità energetiche ■ 5 unità minerarie (consumo minimo se si esclude quello di 100 abitanti che è di una unità energetica, ma che noi portiamo in questo momento ■ 5 unità...), di conseguenza, essendo la risoluzione delle costruzioni di 1 milione ne segue che 5 moltiplicato 1 milione fa 5 milioni e siccome 1000 unità, ■ia energetiche che minerarie, occupano 1 kmq, il massimo valore che queste grandezze possono assumere è (virtualmente) 256 miliardi (1000 × 256), che diviso per 5 milioni fa 51200, ovvero una cifra che può essere agevolmente contenuta in due byte (fine del delirio). Le risorse alimentari, invece, possono avere una risoluzione più bassa (che in questo caso diventa un vantaggio) e precisamente di 100 milioni di unità (infatti questa è la quantità consumata da 100 milioni di abitanti...). Il massimo valore che assume la grandezza in questione ■ identico a quello delle «energie» o dei «minerali» ■ quindi dividendo per 100 milioni si ottiene 2560, il che, come i più attenti avranno notato, non ci permette di guadagnare spazio perché

2560 deve essere per forza di cose memorizzato in più di un byte e cioè in due, ovviamente «comodi comodi». Discorso molto simile anche per la «quantità di armamenti»: risoluzione di 10 milioni di unità dovuta alla produzione di 1 milione di centri costruttivi. Lo spazio occupato da 1000 unità in questo caso è di 10 kmq, quindi 100 unità per kmq. Moltiplichiamo 256 per 100 e poi dividiamo per 10 e otteniamo 2560 e cioè 2 byte di occupazione. L'informazione «prodotti giornalieri» deve essere memorizzata per ogni tipo di risorsa e, senza effettuare calcoli «semplificatori», verrà memorizzata in 2 byte per ogni grandezza (totale ■ byte). Le informazioni «tipo pianeta» e «tipo popolazione» si possono comprimere (si fa per dire) in un byte. I «tipi di energie, minerali e alimenti posseduti (all'interno del pianeta)» vanno memorizzati ciascuno in un byte (quindi al massimo possiamo avere 255 tipi per ogni elemento naturale... forse sono pochi). Facciamo pure la somma dei byte necessari alla memorizzazione di tutte le informazioni di un pianeta e otteniamo 32 byte, che diventano 19 se il pianeta è del tipo «2)» e ■ se è del tipo «3)», in quanto se è del tipo «2)» è sufficiente memorizzare la «superficie disponibile» (1 byte), le «costruzioni possedute» (che in questo caso possono essere solo quelle di produzione energetica, mineraria e militare, quindi 3 byte), i «prodotti giornalieri» (che possono essere di soli tre tipi... 6 byte), le «risorse» (ancora 6 byte), il «tipo di pianeta» (1 byte) e infine le «risorse possedute all'interno del pianeta» (2 byte). Se è del tipo «3)» chiaramente si possono eliminare ancora altri «byte»; restano infatti solo la «superficie disponibile», le «risorse possedute» e il tipo di pianeta. Calcoliamoci pure quant'è la totalità di byte necessaria per immagazzinare la struttura dati di tutti i pianeti: 20 × 32 più 18 × 19 più 21 × 8, totale 1150 byte!

Mi accorgo solo ora che abbiamo dimenticato di dedicare «due» byte per la posizione del pianeta nella mappa... 2 × 59 = 118 (più 1150 fanno 1268 byte).

Finalmente sappiamo su quali dati dobbiamo lavorare. Il problema è ora quello di realizzare la routine. I calcoli da fare sono:

— incremento della popolazione (può essere anche negativo...)
— risorse prodotte
— risorse utilizzate.

Chiaramente devono essere eseguiti basandosi sulle considerazioni fatte la volta scorsa a proposito dei consumi e dei prodotti. Chi si sente in grado di realizzare questa routine? Probabilmente i lettori più fedeli sono già lì che scalpitano e non vedono l'ora di entrare in azione e se ciò fosse vero sarebbe una gran bella cosa... Provate a mandarmi il vostro lavoro (basta anche solo il listato). È ovvio che il programmino sarà scritto utilizzando LABEL al posto di locazioni di memoria specifiche (in modo da poterlo rilocare ovunque). Nella prossima puntata esamineremo qualche altra routine (non vi anticipo nulla). Per quest'anno può bastare...

Scuola di videogame

A partire da questa puntata la rubrica «Scuola di videogame» ospiterà una serie di lezioni relative alla realizzazione di un videogame tipo «GHOSTS'N'GOBLINS», ovvero quel genere di videogiochi che ci mettono, tramite joystick, al comando di un personaggio che si muove lungo un fondale a scrolling (nel nostro caso «a spinta») orizzontale e che deve affrontare, con l'aiuto della sua brava arma, una lunga serie di ostacoli e mostriciattoli, al fine di raggiungere una meta prestabilita.

Il videogioco (del quale potete osservare alcune foto in queste pagine... foto 7, 8, 9, 10 e 11) è stato da me parzialmente realizzato molto tempo fa ■ poi abbandonato a se stesso. Al momento è in versione demo. Il nostro scopo sarà quello di osservare il lavoro fatto finora

Foto 7

Foto ■

Foto 9

■ ovviamente di continuarlo fino al termine.

La trama del gioco è questa: «Una incredibile flotta di esseri malvagi ha invaso la terra ■ ha ridotto gli esseri umani in condizioni di schiavitù. Una splendida ragazza, durante un esperimento di laboratorio atto ad aumentare le capacità mentali umane, rimane vittima di un incidente che altera la sua fisionomia trasformandola per metà in un terribile mostro. Fortunatamente l'esperimento comporta anche dei cambiamenti vantaggiosi ed ora essa ha il potere di sprigionare dalla mente un terribile flusso di energia mortale. Decide perciò di sfruttare questa sua capacità ■ fin di bene ■ indossato un abito da classico supereroe ■ propone di sconfiggere tutti i cattivoni...». Come potete notare il concetto di gioco è estremamente originale ■ ha richiesto mesi ■ mesi di studio... chiaramente sto scherzando, ma in fondo quello che conta è avere un pupazzo da muovere per lo schermo che «scrolla».

Foto 10

Foto 11

Analizziamo innanzi tutto la struttura generale del programma. La routine dei movimenti del giocatore permette di:
— saltare
— abbassarsi
— camminare a destra e sinistra
— «sparare».

Per aumentare la giocabilità permettiamo al giocatore di direzionare il personaggio (a destra o a sinistra) anche quando siamo a salto inoltrato, ovvero mentre il «protagonista» (che da adesso in poi chiameremo «ragazza») è in «volo». Ovviamente è possibile sparare in qualunque momento, quindi anche in abbinamento agli altri comandi.

Quando la ragazza raggiunge il centro dello schermo si attiva la routine di scrolling e la ragazza (che è costituita da due sprite) non si muove più (chiaramente rispetto al monitor). Il raggio d'azione dell'arma di difesa è limitato e quindi bisogna sparare a distanza ravvicinata e con prudenza, in quanto ad ogni colpo viene decrementata l'energia mentale della ragazza ■ se quest'ultima finisce non possiamo più sparare. Il contatto con gli sprite dei nemici comporta il decremento dell'energia che se giunge a zero comporta la fine della partita. La partita può finire anche prima se la ragazza cade in un burrone. Il gioco è strutturato in livelli che si differenziano l'uno dall'altro dallo stile grafico del paesaggio e dai mostri. Durante il cammino è anche possibile incontrare dei bonus che ci incrementano l'energia mentale o quella fisica, oppure il punteggio.

Ci servono una routine di scrolling, una per gestire i comandi da joystick, una per muovere i cattivi ■ tutte le altre routine di contorno per controllare le collisioni, la fine del gioco, la musica, la presentazione, la fine di un livello, i bonus e i punteggi. Buona befana!

Megaposta

Questo mese la megaposta comprende solo dei ringraziamenti ad alcuni lettori. Grazie ■ Nicola Colella (Bari), ■ Claudio Parmigiani (Brescia) ■ ad Emiliano D'Onofrio (Roma).

di Marco Pesce

Un videogioco tutto nostro

Come avrete notato il compito di definire la struttura di un programma-gioco di tale complessità è tutt'altro che semplice ed occorre quindi un po' di comprensione per il fatto che ci si sta spendendo tutto questo tempo... e ancora non abbiamo finito... Tenete duro!

Nelle puntate precedenti abbiamo parlato (anche) di come si presentano i menu al cospetto del giocatore... come realizzarli? La foto 1 mostra un miglioramento (almeno spero) nella grafica dei pannelli. La foto 2 è identica alla prima, ma ha uno schema colore diverso. La grafica è in multicolor modo carattere e quindi possiamo utilizzare quattro colori per ogni riquadro di 8×8 pixel. Ci occorre una routine che sia in grado di stampare le cornici rettangolari che potete osservare in foto 3 e che identificano un particolare settore di opzioni (in foto 1). Facendo il confronto con la foto 1 del numero di novembre si nota che c'è stata una separazione grafica dei comandi, la gestione della mappa è racchiusa nel pannello in alto a destra, mentre gli altri quattro comandi in quello inferiore (potere, viaggi, guerre e controllo). Un'altra routine fondamentale è quella che deve identificare la posizione del puntatore per azionare, in caso di pressione del tasto di fuoco, il comando relativo. È sufficiente realizzare, per ogni pannello, una lista delle posizioni dei comandi nella schermo per risolvere il problema. Ovviamente queste posizioni sono «relative» in quanto devono valere per i pannelli sia del primo che del secondo giocatore. A proposito... abbiamo detto la volta scorsa che i due giocatori comandano civiltà strutturate in maniera totalmente diversa e, paradosso dei paradossi, la «struttura» dei vari menu è identica... non mi sembra il caso di stare a sottilizzare! (o è il caso?).

Parliamo di dissolvenze: i pannelli sono essenzialmente formati da tre colori più il colore di fondo (nero). Ci sono tuttavia degli elementi che hanno colori estranei a quella che è la palette principale (bianco rosa, rosso nella foto1 o bianco, azzurro, blu nella foto 2). Per effettuare una dissolvenza (sia in «entrata» che in «uscita») occorre gradualmente «accendere» o «spegnere» i colori dello schermo. Per effettuare questa operazione in modo semplice l'ideale sarebbe poter agire su determinati registri di input/output dell'hardware grafico (vedi Amiga) e in parte ciò è possibile (per due colori). Resta comunque uno dei colori, che sarebbe poi quello ridefinibile per ogni spazio-carattere, che non è (per nostra fortuna... in questo caso sfortuna) identificato da un registro, ma dalla mappa colore che, come tutti sanno, è di 1 kbyte. Per accendere o spegnere anche questo colore occorre alterare detta mappa e ciò non comporta grandi problemi se non quello relativo al tempo che questa operazione ruba. Il problema nasce dal momento che ci sono i famosi elementi estranei alla palette di base e che a rigor di precisione andrebbero dissolti per conto proprio e quindi complicando «l'alterazione» della mappa colore; e noi invece siamo furbi e questi elementi li facciamo «apparire» a dissolvenza avvenuta. La dissolvenza consiste in definitiva nel passare gradualmente dal nero assoluto al colore e viceversa, per gli elementi aventi palette generale, e di passare dal nero al colore (non gradualmente) per gli elementi colorati «in modo più vario». Per concludere il discorso sulle palette ricordiamo che la palette dei pannelli è diversa da quella dei sottogiochi arcade e quindi la suddivisione a raster (mi sembra di averlo già accennato) dovrà alterare anche i registri colore. Ehi! ma i giochi arcade quali sono? Già nella fretta ce li siamo dimenticati o meglio li abbiamo trascurati. Ricordiamoci che l'azione dinamica del gioco si ritroverà nelle battaglie ■ quindi i giochi arcade saranno dedicati proprio alle battaglie, quelle classiche con astronavi e navicelle di ogni tipo... ma noi non vogliamo limitarci ad un semplice spara e fuggi e quindi decidiamo che combattimenti potranno essere realizzati sia in cielo che in terra... il gioco terrestre ovviamente sarà un parente stretto di ghosts'n'goblins (anche questo l'avevo già accennato). Lo spaziale probabilmente sarà a scorrimento orizzontale (alla faccia dei cloni di R-Type) e comprenderà la distruzione, oltre che delle astronavi avversarie, anche del fondale (quindi edifici ■ installazioni di ogni tipo). La scelta dello scorrimento orizzontale ovviamente non è casuale ma è fatta per economizzare il codice necessario per lo scrolling, che così facendo diventa unico per entrambi i giochi; attendo vostri pareri in merito.

Diamo un'occhiata alla struttura generale del gioco. Le routine che gestisce la vita dei pianeti lavora sulla struttura dati dei pianeti sia in input che in output ovvero legge i dati in essa contenuti, effettua alcuni calcoli e quindi modifica i dati stessi.

La routine di gestione dei menu usufruisce di una subroutine che effettua il controllo del puntatore ■ di un'altra sub che identifica la posizione dello stesso ■ infine di un'altra sub che effettua la stampa dei rettangoli-cornice dei pannelli, questo per tutti i menu che presenterà sullo schermo. Le operazioni effettive tuttavia sono gestite da routine dedicate:
— la gestione della mappa permette di agevolare la selezione di un pianeta e deve fornire come parametro finale appunto un pianeta, che poi servirà alla routine di «controllo»;
— il pannello del potere non è interattivo ■ ■ limita ■ mostrare lo stato di alcuni parametri;
— il pannello della situazione viaggi è interattivo e quindi deve seguire le direttive impostate dalle scelte del giocatore;
— il pannello situazione guerre è simile, per impostazione, al precedente;
— il pannello controllo fa accedere a nuovi «sub-pannelli» (situazione generale, strutture, trasferimenti, attacco).

La gestione dei viaggi è basata sulla seguente struttura dati:
— pianeta sorgente
— pianeta destinazione
— tempo di viaggio
— materiale trasportato
— tipo di viaggio (commerciale o di attacco).

Ogni viaggio è rappresentato in una lista da questo insieme di dati ■ la sub si occupa di decrementare il «tempo di viaggio» ■ di controllare quindi se il materiale è giunto a destinazione. Una volta terminato il viaggio il materiale viene aggiunto nelle risorse del pianeta ■ il blocco dati viene cancellato dalla lista.

La gestione delle guerre è anch'essa basata su una lista di dati composta come segue:
— pianeta attaccato
— quantità di forze nemiche
— potenza forze nemiche
— quantità forze del pianeta
— potenza forze del pianeta.

Nonostante ci sia una percentuale di fortuna, il combattimento si basa essenzialmente su questi dati ed è in base ad essi quindi che si determina la sconfitta di uno dei due gruppi di forze. Ciò naturalmente vale solo se la battaglia viene gestita dal computer, mentre se il giocatore decide di prendere parte all'a-

Foto 1

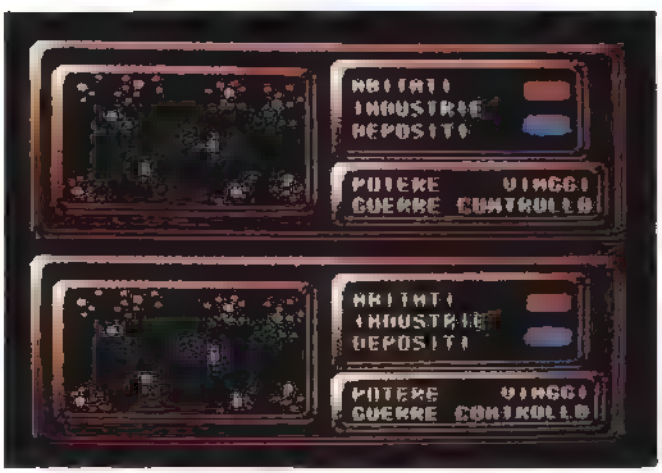


Foto 2

Foto 3

zione (con il gioco arcade) le cose andranno diversamente e l'esito dipenderà in buona parte anche dalla sua bravura. Una volta stabilito il vincitore sorge un problema: se il vincitore è il giocatore attaccante, come sopravviveranno i piloti delle apparecchiature belliche nell'attesa dei sostegni per ricostruire il pianeta? Infatti se è vero che tutte le apparecchiature ■ le risorse di una civiltà non vanno bene per l'altra è anche vero che quei poveracci hanno vinto, ma si ritrovano senza nessuna possibilità di ristrutturare il pianeta distrutto. Per risolvere questo problema abbiamo due possibilità; la prima consiste nel fornire i piloti di una certa autonomia, nell'attesa dei soccorsi, che quindi saranno in grado di «tirare avanti», mentre la seconda consiste nel caricare sulle navi da guerra anche i materiali per la ricostruzione... meglio la prima soluzione. La realizzazione di ciò si traduce nell'aggiungere alla struttura dati del pianeta una certa quantità di energia e di alimenti proporzionata al numero di astronavi rimaste in vita, non appena il pianeta diventa possesso degli attaccanti. È evidente che il problema non esiste se gli attaccanti falliscono la loro missione...

La routine degli imprevisti agisce casualmente ed è comunque abbastanza raro il suo intervento.

La routine delle scoperte agisce ugualmente in modo casuale, ma le probabilità aumentano all'aumentare dei centri di ricerca installati sulla totalità dei pianeti a disposizione del giocatore.

La routine del modo soccorso usufruisce di una lista-dati simile ■ quelle utilizzate per la gestione dei viaggi e delle guerre. In essa è contenuta tutta la lista di comunicazione che bisognerà effettuare una volta attivata detta routine. Ovviamente la lista verrà creata da altre routine quali quella degli imprevisti o quella del calcolo della vita dei pianeti... a proposito: detta routine dovrà fornire anche questo genere di informazioni ovvero ogni qualvolta si ritrova a dover effettuare calcoli con dati insufficienti aggiunge una struttura dati alla lista di comunicazione della routine dei «Soccorsi». Ho scoperto tra l'altro che nella struttura dati dei pianeti c'è una grave mancanza: non sono indicate le installazioni attivate tra quelle costruite (è infatti possibile, come detto, disattivare e riattivare le strutture in nostro possesso); occorre quindi aggiungere tale blocco di dati relativo ad ogni pianeta e ad ogni installazione! Mi scuso con tutti i lettori che stanno lavorando sull'incriminata routine.

Tutti i dati vengono aggiornati in tempo reale e anche i display dei menu sono aggiornati con la stessa modalità. Le routine più importanti sono queste.

Ed ora diamo un'occhiata all'interazione tra le varie routine. Lo schema di figura 1 riepiloga la struttura del programma. Le routine dei menu devono girare in IRQ in quanto anch'esse non possono attendere che la routine principale finisca di effettuare i suoi calcoli, in quanto devono rispondere in tempo reale alle richieste del giocatore. Di conseguenza la gestione dei menu avviene in modo un po' particolare e cioè ad ogni passaggio del raster ci sarà l'eventuale spostamento del puntatore, quindi il controllo della pressione del tasto di fuoco e l'eventuale attivazione di un sub menu o comunque di un comando, ma terminato detto controllo si uscirà dalla sezione menu e si eseguiranno le altre routine IRQ, quindi ■ ritornerà al programma principale. A tal proposito è necessario che la routine dei menu si «ricordi» del... menu attivo, in modo così da poterlo richiamare ad ogni IRQ. Occorre notare che i menu dovrebbero alterare in alcuni casi delle strutture dati sulle quali probabilmente il programma principale sta lavorando, con tutte le conseguenze immaginabili. È necessario quindi sviluppare un sistema di comunicazione tra la routine IRQ e la routine MON-IRQ. Detta comunicazione avverrà tramite dei buffer che verranno alterati normalmente dalla routine IRQ ■ verranno «caricati», al momento opportuno, dal programma principale (routine MON-IRQ) nella loro «posizione» reale.

Nonostante i progressi, nella struttura di gioco ci sono probabilmente ancora numerosi nei, che non tarderanno ■ saltar fuori durante un successivo sviluppo. Vi invito perciò a testare la struttura del programma e ■ farmi eventualmente pervenire il vostro parere. A presto.

Figura 1

Schema della struttura del programma

Menu principale (funziona in IRQ)
— Gestione del cursore, della mappa e della «stampa-rettangoli»;
— se si seleziona «POTERE» si attiva la subroutine di visualizzazione del pannello relativo;
— se si seleziona «VIAGGI» si attiva la sub del pannello viaggi;
— se si seleziona «GUERRE» si attiva la sub del pannello che permette di disattivare una o più guerre e quindi si attiva il viaggio di ritorno delle truppe oppure permette di visualizzare il pannello dello stato del pianeta oppure permette di inviare rinforzi e quindi ci sarà la gestione del pannello dei trasferimenti oppure c'è la possibilità di partecipare all'azione e quindi ci sarà l'attivazione di un sub-gioco;
— se si seleziona «CONTROLLO» si deve ancora selezionare un pianeta tra quelli attualmente visualizzati sul monitor e quindi si attiva la sub del pannello di scelta del tipo di controllo che può essere:
1) situazione generale
2) strutture
3) trasferimenti
4) attacco
quindi...
1) visualizza il pannello interattivo
2) permette la scelta tra:
— costruire nuove strutture
— disattivare strutture
— riattivare strutture
— distruggere strutture
quindi gestisce il pannello di selezione tipo e di selezione quantità;
3) gestisce il pannello dei trasferimenti (viaggi) e attiva quindi un viaggio aggiungendo una struttura dati alla lista-viaggi;
4) gestisce il pannello ■ permette di aggiungere una struttura dati alla lista-guerre. In più attiva il pannello «modo-soccorsi» del giocatore avversario.

«Altre» routine IRQ

— routine musicale
— routine dissolvenze

Routine non IRQ

— gestione della PRESENTAZIONE;
— calcolo struttura dati dei pianeti e eventuale aggiunta alla struttura dati del «modo-soccorso» di un «imprevisto» nel caso si verificasse una complicazione nei calcoli (ad esempio: un pianeta ha finito lo spazio a disposizione);
— controllo viaggi;
— controllo guerre e quindi alterazione delle caratteristiche dei pianeti in guerra;
— controllo imprevisti e quindi eventuali aggiunte di dati alla lista del «modo-soccorso»;
— controllo invenzioni;
— se c'è stata una modifica (da parte dei menu) nel buffer di comunicazione con l'IRQ, trasferimento dei nuovi dati nelle opportune tabelle.

Scuola di videogame

Ci eravamo lasciati con l'elenco delle routine che ci servivano per il gioco. Dunque le routine di scrolling ■ di gestione dei comandi da joystick sono già pronte e così è anche per la mappa del primo livello del fondale, per il set di caratteri ridefinito utilizzato da detto mappa, per quello del pannello dei punteggi e per gli sprite relativi ai movimenti del personaggio principale. Nello stato in cui si trova al momento, il programma è fornito anche di una provvisoria presentazione, che consiste in una animazione avente come protagonista la ragazza ■ nella visualizzazione della schermata con il titolo del gioco. Segue il caricamento di altri file ■ finalmente possiamo «giocare». Il pupazzo può agevolmente muoversi lungo un fondale a scrolling orizzontale e può anche sparare, ma tutto quanto è in grado di fare ■ rivela perfettamente inutile in quanto non ci sono nemici ad affrontarlo e anche raggiunta la fine del livello non succede assolutamente nulla. La foto 6 mostra il vecchio editor di sprite in funzione mentre le foto 7 ■ 8 sono tratte dal Kwik-Paint, il programma grafico che utilizzo per disegnare schermate caratteri ■ sprite. Intendiamoci, il programma permette solo di disegnare una schermata grafica. È grazie ■ miei convertitori che poi vengono fuori i caratteri ■ gli sprite.

Esaminiamo la routine di scrolling. Essa è del tipo «niente compressioni» in quanto non utilizza i famosi sistemi tipo S.E.U.C.K. per compattare i dati relativi allo sfondo (che lo rendono più ampio, ma anche più monotono) e quindi occupa una quantità di RAM, a dir poco, notevole, ma in compenso permette di realizzare qualunque tipo di fondale (per il megagame non avremo questa possibilità tipo «a tutta RAM» per ovvi motivi «economici»). Inconveniente della questione: ogni livello viene, anzi deve, essere caricato da disco. Per ogni livello, oltre alla mappa, occorrerà caricare anche un nuovo set di caratteri e un nuovo set di sprite per i nemici. Torniamo allo scrolling. Se non è compattato come è memorizzato? Nel più semplice dei modi: un carattere dopo l'altro. L'ingrato compito di disegnare la mappa è stato affidato al mio schiavetto personale... scherzo; per l'occasione ho realizzato un programma che mi permetteva di disegnare sullo schermo con il set di caratteri del fondale e quindi memorizzava gli screen. Con esso è possibile definire dei pennelli (per le figure ricorrenti) composti da più caratteri e quindi utilizzarli ogni qualvolta serve quel determinato insieme di caratteri. Altra possibilità quella di visualizzare la mappa ■ scorrimento e quindi fermarsi ■ ricominciare a disegnare da quel punto (per le correzioni). Le foto 4 e 5 mostrano il succitato programma all'opera sulla mappa del primo livello; i caratteri apparentemente pasticciati, che vedete nella parte bassa dello schermo, sono la tavolozza dalla quale è possibile catturare un carattere per il disegno. Il listato di figura 2 è quello relativo al codice assembly dello scrolling; per semplificare il compito di trascrittura (a chiunque voglia sperimentare il funzionamento di questa routine) viene proposto anche il listato esadecimale (figura 3). Il listato assembly è notevolmente intricato, non tanto per la sua complessità quanto per il fatto che è stato scritto direttamente con il monitor-assembly, quindi ad ogni minima correzione bisognava fare i conti con la distribuzione del codice. Inutile farvi notare che scrivere direttamente in questo modo anziché utilizzare un vero Assembler, con tanto di label, è il metodo più rapido per finire in manicomio. La routine si attiva con:

SYS 28784 ($7070)

imposta quindi la prima routine IRQ che è memorizzata a partire da $7098 ■ un'interruzione raster alla posizione $F1 (241), scelta comunque a caso; inoltre salta a $704E, dove è memorizzata una routine che effettua con copia dei registri di input-output del VIC II a partire dalla locazione $7200 ■ quindi fino alla locazione $7722E. Ciò è utile in quanto

Foto 4

Foto ■

la nuova mappa di registri è ora quella che va da $7200 ■ $722.

E questo per fare in modo che sia la routine IRQ a gestire l'I/O con il processore grafico. ▮ tutto si traduce nell'assenza di sfarfallamenti. In pratica noi utilizzeremo la nuova mappa come se si trattasse di quella originale e la routine IRQ penserà ad effettuare il «trasferimento». Terminata la «copia» troviamo un «RTS». La nostra attenzione ora deve andare sulla routine IRQ ($7098). Le prime due istruzioni effettuano un'operazione apparentemente inutile che in-

Foto 6

Foto 7

Foto ■

```
:71C0 00 00 FF FF 00 00 FF FF @@..@@..
:7000 A9 27 8D 17 70 A9 04 8D ⌀'MW.⌀DM
:7008 18 70 A9 5F 85 FB A9 7A X.⌀ E.⌀.
:7010 85 FC A0 00 B1 FB 8D A7 E. @┴.M┤
:7018 06 C8 C0 11 F0 11 18 AD FH─Q.QX└
:7020 17 70 69 28 8D 17 70 90 W..(MW.P
:7028 EB EE 18 70 4C 14 70 A9 ..x.LT.⌀
:7030 D7 8D 16 D0 60 A2 10 BD WMVP.▄P┘
:7038 FF 71 9D FF CF CA D0 F7 ..┤.DJP.
:7040 A2 0F BD 19 72 9D 19 D0 ▄D┘Y.┤YP
:7048 CA D0 F7 4C BC FE A2 2F JP.L┴.▄/
:7050 BD FF CF 9D FF 71 CA D0 ┘.DI..JP
:7058 F7 60 00 00 00 00 00 00 ..@@@@@@
:7060 00 00 00 00 00 00 00 00 @@@@@@@@
:7068 00 00 00 00 00 EA EA EA @@@@@...
:7070 78 A9 7F 8D 0D DC A9 98 .⌀.MM▫⌀X
:7078 A2 70 8D 14 03 8E 15 03 ▬.MTCNUC
:7080 EA A9 F1 8D 12 D0 AD 11 .⌀.MRP┘Q
:7088 D0 29 7F 8D 11 D0 A9 01 P).MQP⌀A
:7090 8D 1A D0 58 4C 4E 70 15 MZPXLN.U
:7098 AD 97 70 8D 18 D0 EA EA ┘W.MXP..
:70A0 A9 FF 8D 19 D0 A9 B2 A0 ⌀.MYP⌀┬
:70A8 71 8D 14 03 8C 15 03 A9 .MTCLUC⌀
:70B0 F8 8D 12 D0 AD 11 D0 29 .MRP┘QP)
:70B8 7F 8D 11 D0 A9 D0 8D 16 .MQP⌀PMV
:70C0 D0 4C 35 70 EA EA EA EA PL5.....
:70C8 EA EA EA EA D0 00 EA EA ....P@..
:70D0 A9 FF 8D 19 D0 A9 98 A0 ⌀.MYP⌀X
:70D8 70 8D 14 03 8C 15 03 A9 .MTCLUC⌀
```

```
:70E0 BA BD 12 D0 AD 11 D0 29 ┌MRP┘QP)
:70E8 7F 8D 11 D0 AD CD 70 F0 .MQP┴M..
:70F0 1A A9 00 8D CD 70 AD CC Z⌀@MM.┘L
:70F8 70 29 07 F0 06 CE CC 70 .)G.FNL.
:7100 4C 0B 71 A9 D7 8D CC 70 LK.⌀WML.
:7108 20 84 71 AD CC 70 8D 16  D.┘L.MV
:7110 D0 4C 31 EA A2 00 BD 01 PL1.▄D┘A
:7118 04 9D 00 04 BD 29 04 9D D(@D┘)D1
:7120 28 04 BD 51 04 9D 50 04 (D┘QDPPD
:7128 BD 79 04 9D 78 04 BD A1 ┘.D1.D┘▮
:7130 04 9D A0 04 BD C9 04 9D D1 D┘1D1
:7138 C8 04 BD F1 04 9D F0 04 HD┘.D1.D
:7140 BD 19 05 9D 18 05 BD 41 ┘YE1XE┘A
:7148 05 9D 40 05 BD 69 05 9D E1@E┘.E1
:7150 68 05 BD 91 05 9D 90 05 .E┘QE1PE
:7158 BD B9 05 9D B8 05 BD E1 ┘▬E1╕E┘.
:7160 05 9D E0 05 BD 09 06 9D E1.E┘IF1
:7168 08 06 BD 31 06 9D 30 06 HF┘1F0F
:7170 BD 59 06 9D 58 06 BD B1 ┘YF1XF┘A
:7178 06 9D B0 06 EB E0 27 D0 F1┤F..'P
:7180 95 4C 00 70 18 AD 0B 70 ULe.X⌀..
:7188 69 11 8D 0B 70 90 0A EE .DMF.PJ.
:7190 0F 70 AD 0F 70 C9 60 F0 O.⌀O.I..
:7198 03 4C 14 71 A9 FF 8D 64 CLT.⌀.M.
:71A0 70 4C 14 71 A9 40 8D 0F .LT.⌀@MO
:71A8 70 A9 00 8D 0B 70 4C 70 .⌀@MK.L.
:71B0 70 1F AD B1 71 8D 1B D0 ._ └.MXP
:71B8 4C D0 70 EA EA EA EA EA LP......
```

Figura 3

vece serve per cambiare i set di caratteri che la parte di schermo di prossima visualizzazione dovrà utilizzare (e cioè la parte del pannello dei punteggi). Segue l'impostazione della nuova (la seconda) routine IRQ quindi della nuova posizione del raster IRQ ($FB, 251) ■ infine l'azzeramento della posizione X dello scrolling fine (locazione $D016). Si salta dunque a $7035 dove troviamo la famosa routine del «trasferimento» nei registri di I/O quindi il salto a $FEBC. Giunti alla posizione del raster $FB ci sarà la chiamata della routine memorizzata ■ partire da $71B2. In pratica troviamo le stesse due istruzioni che iniziavano l'altra routine IRQ e che servivano a modificare il set di caratteri da utilizzare; in questo caso il set sarà quello della mappa dello scrolling. La routine segue a $70D0 ■ anche qui troviamo l'impostazione della nuova routine IRQ (che in questo caso è la prima) e della nuova posizione raster ($BA, 186) per la prossima IRQ. A questo punto c'è una istruzione importante: quella che controlla se dobbiamo effettuare uno scrolling di un pixel. Il controllo si basa sulla verifica del contenuto della locazione $70CD (28877); se in questa locazione c'è un valore diverso da zero ci sarà scrolling, altrimenti si salta a $710B e si ritorna al non-IRQ. Detta locazione servirà quindi per effettuare il collegamento tra il resto del programma (non-IRQ) ■ appunto la routine di scrolling; ogni volta che si «pokerà» in questa locazione un valore diverso da zero avremo il nostro scrolling di un pixel.

Vediamo cosa succede in caso di scrolling. Innanzi tutto si azzera la locazione $70CD per evitare che alla prossima IRQ ci sia un nuovo scrolling non voluto. Si testa quindi il valore della locazione $70CC (che contiene il valore di scrolling fine della posizione X) per verificare che non si sia usciti dal range di scrolling fine (0-7).

Quindi, se non si è usciti, detta locazione viene DECrementata e si salta quindi a D710B.

Se invece siamo usciti ci sarà un salto alla locazione $7103, che «azzera» la locazione $70CC e quindi effettua lo scrolling grosso di tutti i caratteri dello schermo interessati all'area della mappa di gioco.

Lasciamoci con questa routine in sospeso... Il prossimo mese la esamineremo.

```
.7000 A9 27     LDA #$27
.7002 8D 17 70  STA $7017
.7005 A9 04     LDA #$04
.7007 8D 18 70  STA $7018
.700A A9 5F     LDA #$5F
.700C 85 FB     STA $FB
.700E A9 7A     LDA #$7A
.7010 85 FC     STA $FC
.7012 A0 00     LDY #$00
.7014 B1 FB     LDA ($FB),Y
.7016 8D A7 06  STA $06A7
.7019 C8        INY
.701A C0 11     CPY #$11
.701C F0 11     BEQ $702F
.701E 18        CLC
.701F AD 17 70  LDA $7017
.7022 69 28     ADC #$28
.7024 8D 17 70  STA $7017
.7027 90 E8     BCC $7014
.7029 EE 18 70  INC $7018
.702C 4C 14 70  JMP $7014
.702F A9 D7     LDA #$D7
.7031 8D 16 D0  STA $D016
.7034 60        RTS
.7035 A2 10     LDX #$10
.7037 BD FF 71  LDA $71FF,X
.703A 9D FF CF  STA $CFFF,X
.703D CA        DEX
.703E D0 F7     BNE $7037
.7040 A2 0F     LDX #$0F
.7042 BD 19 72  LDA $7219,X
.7045 9D 19 D0  STA $D019,X
.7048 CA        DEX
.7049 D0 F7     BNE $7042
.704B 4C BC FE  JMP $FEBC
.704E A2 2F     LDX #$2F
.7050 BD FF CF  LDA $CFFF,X
.7053 9D FF 71  STA $71FF,X
.7056 CA        DEX
.7057 D0 F7     BNE $7050
.7059 60        RTS
.705A 00        BRK
.705B 00        BRK
.705C 00        BRK
.705D 00        BRK
.705E 00        BRK
.705F 00        BRK
.7060 00        BRK
.7061 00        BRK
.7062 00        BRK
.7063 00        BRK
.7064 00        BRK
.7065 00        BRK
.7066 00        BRK
.7067 00        BRK
.7068 00        BRK
.7069 00        BRK
.706A 00        BRK
.706B 00        BRK
.706C 00        BRK
.706D EA        NOP
.706E EA        NOP
.706F EA        NOP
.7070 78        SEI
.7071 A9 7F     LDA #$7F
.7073 8D 0D DC  STA $DC0D
.7076 A9 98     LDA #$98
.7078 A2 70     LDX #$70
.707A 8D 14 03  STA $0314
.707D 8E 15 03  STX $0315
.7080 EA        NOP
.7081 A9 F1     LDA #$F1
.7083 8D 12 D0  STA $D012
.7086 AD 11 D0  LDA $D011
.7089 29 7F     AND #$7F
.708B 8D 11 D0  STA $D011
.708E A9 01     LDA #$01
.7090 8D 1A D0  STA $D01A
.7093 58        CLI
.7094 4C 4F 70  JMP $704E
.7097 AD 97 70  LDA $7097
.709A 8D 18 D0  STA $D018
.709D EA        NOP
.709E EA        NOP
.709F EA        NOP
.70A0 A9 FF     LDA #$FF
.70A2 8D 19 D0  STA $D019
.70A5 A9 B2     LDA #$B2
.70A7 A0 71     LDY #$71
.70A9 8D 14 03  STA $0314
.70AC 8C 15 03  STY $0315
.70AF A9 FB     LDA #$FB
.70B1 8D 12 D0  STA $D012
.70B4 AD 11 D0  LDA $D011
.70B7 29 7F     AND #$7F
.70B9 8D 11 D0  STA $D011
.70BC A9 D0     LDA #$D0
.70BE 8D 16 D0  STA $D016
.70C1 4C 35 70  JMP $7035
.70C4 EA        NOP
.70C5 EA        NOP
.70C6 EA        NOP
.70C7 EA        NOP
.70C8 EA        NOP
.70C9 EA        NOP
.70CA EA        NOP
.70CB EA        NOP
.70CC D0 00     BNE $70CE
.70CE EA        NOP
.70CF EA        NOP
.70D0 A9 FF     LDA #$FF
.70D2 8D 19 D0  STA $D019
.70D5 A9 98     LDA #$98
.70D7 A0 70     LDY #$70
.70D9 8D 14 03  STA $0314
.70DC 8C 15 03  STY $0315
.70DF A9 BA     LDA #$BA
.70E1 8D 12 D0  STA $D012
.70E4 AD 11 D0  LDA $D011
.70E7 29 7F     AND #$7F
.70E9 8D 11 D0  STA $D011
.70EC AD CD 70  LDA $70CD
.70EF F0 1A     BEQ $710B
.70F1 A9 00     LDA #$00
.70F3 8D CD 70  STA $70CD
.70F6 AD CC 70  LDA $70CC
.70F9 29 07     AND #$07
.70FB F0 06     BEQ $7103
.70FD CE CC 70  DEC $70CC
.7100 4C 0B 71  JMP $710B
.7103 A9 D7     LDA #$D7
.7105 8D CC 70  STA $70CC
.7108 20 84 71  JSR $7184
.710B AD CC 70  LDA $70CC
.710E 8D 16 D0  STA $D016
.7111 4C 31 EA  JMP $EA31
.7114 A2 00     LDX #$00
.7116 BD 01 04  LDA $0401,X
.7119 9D 00 04  STA $0400,X
.711C BD 29 04  LDA $0429,X
.711F 9D 28 04  STA $0428,X
.7122 BD 51 04  LDA $0451,X
.7125 9D 50 04  STA $0450,X
.7128 BD 79 04  LDA $0479,X
.712B 9D 78 04  STA $0478,X
.712E BD A1 04  LDA $04A1,X
.7131 9D A0 04  STA $04A0,X
.7134 BD C9 04  LDA $04C9,X
.7137 9D C8 04  STA $04C8,X
.713A BD F1 04  LDA $04F1,X
.713D 9D F0 04  STA $04F0,X
.7140 BD 19 05  LDA $0519,X
.7143 9D 18 05  STA $0518,X
.7146 BD 41 05  LDA $0541,X
.7149 9D 40 05  STA $0540,X
.714C BD 69 05  LDA $0569,X
.714F 9D 68 05  STA $0568,X
.7152 BD 91 05  LDA $0591,X
.7155 9D 90 05  STA $0590,X
.7158 BD B9 05  LDA $05B9,X
.715B 9D B8 05  STA $05B8,X
.715E BD E1 05  LDA $05E1,X
.7161 9D E0 05  STA $05E0,X
.7164 BD 09 06  LDA $0609,X
.7167 9D 08 06  STA $0608,X
.716A BD 31 06  LDA $0631,X
.716D 9D 30 06  STA $0630,X
.7170 BD 59 06  LDA $0659,X
.7173 9D 58 06  STA $0658,X
.7176 BD B1 06  LDA $06B1,X
.7179 9D B0 06  STA $06B0,X
.717C E8        INX
.717D E0 27     CPX #$27
.717F D0 95     BNE $7116
.7181 4C 00 70  JMP $7000
.7184 18        CLC
.7185 AD 0B 70  LDA $700B
.7188 69 11     ADC #$11
.718A 8D 0B 70  STA $700B
.718D 90 0A     BCC $7199
.718F EE 0F 70  INC $700F
.7192 AD 0F 70  LDA $700F
.7195 C9 80     CMP #$80
.7197 F0 03     BEQ $719C
.7199 4C 14 71  JMP $7114
.719C A9 FF     LDA #$FF
.719E 8D 64 70  STA $7064
.71A1 4C 14 71  JMP $7114
.71A4 A9 40     LDA #$40
.71A6 8D 0F 70  STA $700F
.71A9 A9 00     LDA #$00
.71AB 8D 0B 70  STA $700B
.71AE 4C 70 70  JMP $7070
.71B1 1F        ???
.71B2 AD B1 71  LDA $71B1
.71B5 8D 18 D0  STA $D018
.71B8 4C D0 70  JMP $70D0
.71BB EA        NOP
.71BC EA        NOP
.71BD EA        NOP
.71BE EA        NOP
.71BF EA        NOP
```

Figura 2

Megaposta

Caro Marco, sono sempre io, Emiliano. È la terza volta che ti scrivo, ma l'entusiasmo è come quello della prima. Spero solo che non mi annovererai fra quelli che lanciano una pietruzza ogni tanto... la trovata dei due giocatori non è male... però tutta quella mappazza di argomenti riguardanti i pianeti da conquistrare mi sembra esagerata... ulteriori complicazioni sui dati significherebbe troppe scritte e niente gioco... gli screen che mostrano le scene di battaglia dovranno essere a dir poco perfetti... Ed ora scusami, devo dormire, chissà che non sogni il più bello degli spaziali — Starglider II —, ma noi lo supereremo no?....

Emiliano D'Onofrio, Roma

Caro Emiliano, tranquillizzati pure perché non sei certo uno di quelli che «lanciano pietruzze». Innanzi tutto scusami per i tagli alla lettera, ma come avrai capito (e mi rivolgo anche agli altri lettori), se le lettere venissero pubblicate tutte integralmente non ci sarebbe più spazio per la rubrica. Tengo a precisare che la «mappazza» di dati ve la siete cercata voi; inizialmente mi avevano accusato di prendere la cosa troppo alla leggera ed ora... Comunque da quanto ho capito la tua lettera è stata scritta prima dell'uscita del numero di novembre e quindi si basava sulle considerazioni del numero di ottobre; in fondo, una volta spiegato il loro significato, nessuno dei dati è trascurabile. Per le idee che mi proponi... non sono male, ma molte andrebbero a modificare troppo la struttura che abbiamo già creato. Continua così Emiliano!

Caro Marco, è la prima volta che ti scrivo, seguo la tua iniziativa fin dall'inizio con molto interesse e mi complimento per come curi la rubrica. Però da quello che ho letto vedo che incontri alcuni ostacoli, secondo me dovuti alla poca interattività che si ha tra i lettori e la rubrica, dovuta al mezzo di comunicazione: la lettera. Per questo ti vorrei proporre alcune idee per ovviare a questo ostacolo, anche perché ritengo che la rubrica ha tutti i numeri per sfondare ed è un peccato non darle il giusto valore.

Io proporrei di organizzare dei gruppi di lavoro per città, approfittando del fatto che tu hai tutti gli indirizzi di coloro che ti scrivono, proponendo di mettersi in contatto... nulla vieta la partecipazione di persone singole, come ora... poi si potrebbe istituire una linea telefonica diretta del tipo ON-LINE per particolari quesiti e non...

Ciro Cané, Casalnuova (NA)

Sono contento che ci sia gente come te che prende la cosa molto sul serio ■ vorrei poterti accontentare in tutto, ma... Per quanto riguarda i gruppi di lavoro, come idea non è male, ma tutto chiaramente dipende dal fatto che ci siano o meno lettori disposti ad aderire alla tua proposta. La linea telefonica diretta sarebbe una buona idea se solo fosse possibile realizzarla; secondo te chi dovrebbe essere quel disgraziato pronto a rispondere ad ogni richiesta? (vuoi forse farmi passare il resto dei miei giorni attaccato ad un telefono?). Scrivi ancora.

Caro Marco Pesce, spero che in questa megagalattica (è proprio il caso di dirlo dato l'argomento del gioco) lettera troverai qualche spunto interessante (sempre che tu abbia il coraggio di leggerla tutta)... in ogni modo spero che mi indicherai nella Megaposta semplicemente di averla letta, così perlomeno saprò che ti è arrivata ■ non sarò preso dalla tentazione di scriverne un'altra così.

Bruno Saetta, Napoli

Per quanto mi riguarda puoi scrivere anche altre 10 lettere così; l'importante è che i lettori non le leggano... Non preoccuparti, stavo scherzando, ma purtroppo la tua lettera era davvero troppo lunga, comunque, almeno io, l'ho letta tutta e sicuramente ci sono alcune cose buone... ma di musica digitalizzata durante il caricamento non se ne parla proprio. Non perché sia impossibile da realizzare (comunque è quasi impossibile), ma per il fatto che una musica del genere occupa quasi tutta la memoria e quindi, se deve sentirsi durante il caricamento, dove carichiamo il programma se la RAM è già tutta occupata? E ammesso che ci sia a disposizione una parte di RAM ancora libera, lo sai che più cose si fanno durante un caricamento e più si rallenta questa operazione? Una musica digitalizzata occupa «un sacco di tempo» e ci vorrebbero dei secoli per caricare tutto il programma... Comunque non considerarlo un rimprovero, oltretutto hai ammesso di non saper programmare, quindi... A proposito del caricamento da disco: Il gioco sarà esclusivamente per due giocatori e NON ci sarà opzione per una partita uomo-computer e questo perché a mio avviso (e anche ad avviso di altri) in tal caso il Megagame diverrebbe molto noioso. Cosa c'entra questo con il caricamento da disco? Semplice: datosi che non possiamo interrompere in nessun momento la dinamica del gioco, non potranno esistere caricamenti multipli!

Caro Marco, sono Gennaro e mi è subito piaciuta l'idea di costruirmi il digitalizzatore audio perché sono un possessore di un 64 e di un Amiga 500. Per il 64 tutto OK (almeno credo, l'IC ADC 0802 mi deve ancora arrivare), ma per l'Amiga non so assolutamente niente. Per usarlo anche su Amiga come devo fare...

Gennaro Claps, Potenza

Il digitalizzatore deve essere collegato alla porta parallela con otto cavetti che dovrebbero portare il segnale uscente dai piedini dall'11 al 18 dell'ADC 0802 ai rispettivi piedini di detta porta, indicati come «data-bit», quindi collegare l'alimentazione, presa sempre dai piedini di questa porta. Una volta fatto ciò il digitalizzatore funzionerà con tutti i programmi di digitalizzazione audio, tipo l'Audio Master. Chiaramente queste indicazioni sono estremamente superficiali, ma se sei un bravo smanettone elettronico avrai certamente capito cosa devi fare... in ogni modo ti consiglio di aspettare il prossimo numero (su questo non c'è più spazio) per maggiori dettagli.

In breve rispondo anche a Giorgio Zanelladi, Poggibonsi (SI) che mi chiede di preparargli un compattatore di dati per il digitalizzatore. Caro Giorgio, il compattatore non serve... Hai mai sentito parlare di «Digital Voice»? Ne riparliamo sul prossimo numero...

E rispondiamo anche a Lorenzo Ascari di Reggio Emilia. Colui che nel mese di ottobre ha scritto l'articolo sul «digitalizzatore di audio» si chiama (guarda caso) Marco Pesce e i componenti per il circuito li ha «prelevati» da un laboratorio di elettronica digitale... mi informerò e ti farò sapere come reperire l'integrato.

di Marco Pesce

Un videogioco tutto nostro

In questa puntata ho effettuato alcune modifiche al gioco, dopo le numerose discussioni con mio fratello sul fatto che lui preferiva una vista dei combattimenti tridimensionale

Dando uno sguardo alle foto della pagina a fronte potete rendervi conto di come le cose sono cambiate. Con la nostra MEGA-astronave possiamo attaccare un pianeta in modo diretto, quindi non più teleguidando una delle numerose navicelle di contorno da una postazione sicura, ma rischiando realmente la distruzione totale del nostro impero! L'astronave comunque sarà protetta da numerosi scudi difensivi che gli permetteranno di resistere a lungo. Avremo tuttavia dei livelli energetici che verranno alterati da eventuali attacchi ■ comunque, anche se in misura decisamente minore, dall'energia necessaria per gli spostamenti da un pianeta all'altro. È chiaro che sarà anche possibile rifornirsi, commissionando una nave di trasporto di uno dei nostri pianeti per il recapito del materiale. La visuale dei due giocatori sarà sempre quella che vedete nelle foto ■ tutto si svolgerà sul monitor. I menu resteranno più o meno inalterati, almeno per quanto riguarda le loro possibilità. L'attacco di un pianeta si divide in tre fasi:

— combattimento (nello spazio) contro le difese esterne del pianeta;
— combattimento sulla superficie del pianeta;
— combattimento all'interno del pianeta per la distruzione del nucleo centrale.

Una volta distrutto il nucleo centrale il pianeta sarà nostro!

Va da sé che non è così semplice raggiungere tale obiettivo e quindi a volte potrebbe capitare che proprio sul più bello dobbiamo ritirarci (almeno per quello che riguarda la MEGA-astronave) per evitare il danno irreparabile.

Questa modifica cambia molte cose, soprattutto l'impostazione grafica dei menu; niente dissolvenze tra un passaggio di schermo e un altro (dato si che tutto verrà gestito in stile terminale elettronico) ■ niente cornicette decorative. Il problema più grosso sarà quello di visualizzare una convincente sezione in TRE-D (cme avrete capito, abbiamo eliminato le sezioni arcade a scolling orizzontale).

La nostra MEGA-astronave dunque viaggia e per farlo serve un sistema di controllo e nuove strutture dati, come quelle che dovrà stabilire in quale punto della mappa galattica ci troviamo ■ quanta energia possediamo.

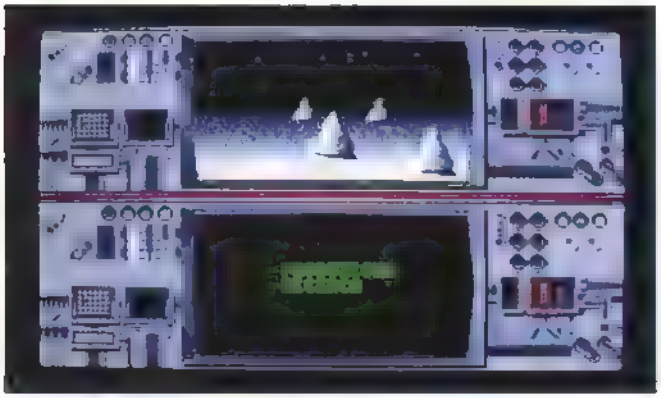
Tutti i gadget che potete ammirare nelle schermate sono per il momento finiti, ma in gran parte dovranno essere trasformati in strumenti realmente utili. Tra l'altro le schermate sono state realizzate con l'Amiga e quindi dovranno essere convertite sul 64 (purtroppo non ho avuto il tempo di farlo, ma cercherò di tirar fuori un pannello più che dignitoso).

Che ne dite di cominciare a pensare ad un nome un po' più adatto per il nostro Megagame? È già, nessuno ci aveva pensato, ma Megagame 64 non si addice granché al tema del gioco; tutto quanto sono riuscito ■ pensare in questi ultimi 10 secondi è «Final War»... pensate anche voi.

Occorre escogitare qualcosa di più complesso per la sezione battaglie. Più complesso dal punto di vista strategico, ovvero non dobbiamo limitarci ■ stabilire quante astronavi mandare e aspettare il resoconto del computer. Abbiamo detto che è possibile partecipare di persona alle battaglie e infatti il nostro è un altro discorso, che riguarda invece la possibilità di non partecipare in modo diretto.

Cosa si vede dal monitor della MEGA-astronave? Un misto di caratteri e sprite, ben congegnato in modo da rendere fluida e bella l'azione. A parole è molto semplice, ma occorre fare numerosi test per realizzare il tutto nel migliore dei modi. È evidente che la maggior parte della RAM sarà occupata dalla grafica e occorreranno delle compressioni ai limiti delle possibilità umane!

Nel prossimo numero cercherò di presentare i primi esperimenti (aspettando le vostre risposte...); speriamo bene. In pratica, approfittando delle ridotte dimensioni dello schermo, sarà possibile utilizzare «sprite veri» per gli oggetti in movimento indipendente dal fondale ■ caratteri per il fondale stesso, quindi ridurre al minimo l'utilizzo di elementi mobili in stile «OUT RUN» o «SPACE HARRIER», che usufruiscono, cioè, di caratteri anche per il movimento di automobili o astronavi.



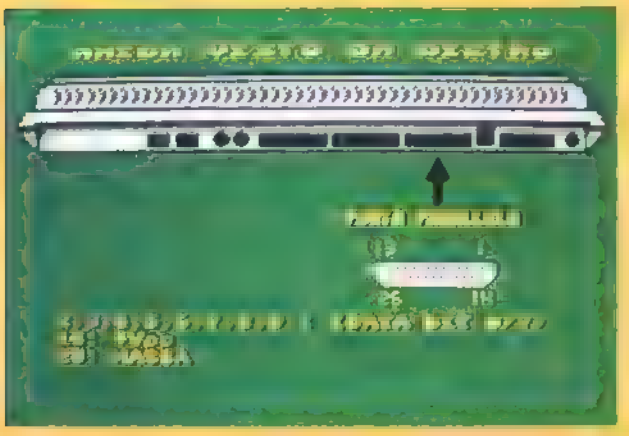
Scuola di videogame

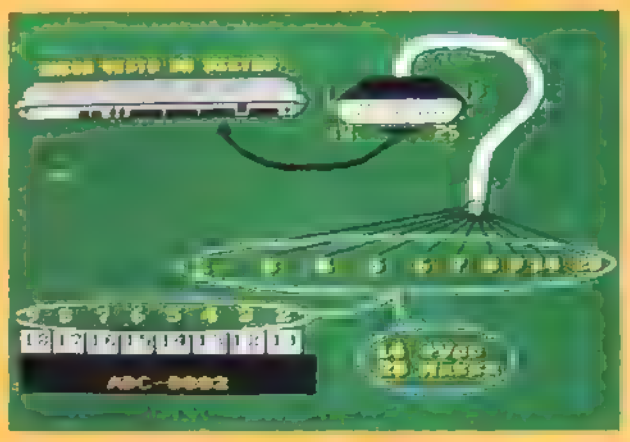
Riprendiamo la spiegazione del listato di scrolling. Alla locazione $7184 troviamo un controllo sullo stato dello scrolling; in pratica viene testata la posizione raggiunta nella tabella dei dati dal puntatore che ne preleva appunto... i dati. Se il byte meno significativo (si tratta di un puntatore a 16 bit) è andato «fuori di carry» ■ incrementa il byte più significativo ■ se è finita la tabella (puntatore= $6000) ■ setta la locazione $7064 a $FF per avvertire la routine non IRQ che il giocatore è giunto alla fine del livello. Ma se la tabella non è ancora finita allora si salta ■ $7114. Prima di saltarci volevo farvi notare la presenza della routine memorizzata a partire da $71A4: ■ la routine di reset che va chiamata prima ancora di attivare tutta l'IRQ-routine ■ che azzera il menzionato puntatore alla locazione $4000 ■ poi salta direttamente a $7070. Saltiamo ora a $7114 e diciamo subito che questo pezzo si occupa dello spostamento di tutti i caratteri dell'area di gioco di un posto verso sinistra, ovvero i 39 caratteri più a destra di ogni riga (per un totale di 17 righe) vengono spostati di uno spazio-carattere ■ la prima colonna di caratteri viene quindi cancellata; in seguito l'ultima colonna viene riempita con nuovi caratteri. Come potete notare, questa sezione è piena di istruzioni simili e i più «impulsivi» penseranno subito che si poteva usare un numero ben più limitato di istruzioni per compiere la stessa operazione; invece ciò non è possibile, in quanto, già in queste condizioni, ci sono problemi di velocità proprio perché spostare 39×17 caratteri in un cinquantesimo di secondo non è uno scherzo ed occorre la maggiore velocità possibile, anche ■ costo di sprecare più memoria. Il massimo della velocità si poteva ottenere utilizzando una «LDA» ■ una «STA» per ogni carattere ma in tal caso si aveva anche il massimo spreco di memoria. Tutto questo trova una spiegazione nel fatto che alcune istruzioni più semplici impiegano meno tempo di altre per essere eseguite ma, proprio perché più semplici, ci allungano le routine.

Da sottolineare il fatto che per ridurre l'impegno del processore era anche possibile eseguire in una seconda pagina video un pezzetto di spostamento per ogni chiamata IRQ e quindi commutare la visuale sulla pagina nascosta, dividendo così per otto «lo sforzo» ma, ancora una volta, sprecando memoria. È bene imparare che è preferibile «sprecare» memoria piuttosto che danneggiare il vostro programma con uno scrolling difettoso. Proseguiamo l'esame della routine; le 17 righe sono suddivise in 17 sezioni di «LDA-STA», che spostano ognuna 39 caratteri, il tutto per l'appunto in 39 passare. Le istruzioni memorizzare in &717D si occupano del conto delle passate ■ se è giunto al termine si salta a $7000. È la subroutine dell'aggiunta, cioè quella che prende, dal blocco di dati elaborato con l'editor di mappa, i caratteri da stampare nella colonna che si è liberata con lo scroll grosso ■ che ora è nascosta dal bordo dello schermo (che è stato allargato). Le prime quattro istruzioni, bisogna ammetterlo, sono piuttosto bizzarre; vanno ad alterare il contenuto della $7017 ■ della $7018, apparentemente senza alcun motivo valido, ma se continuiamo nella nostra lettura ci accorgiamo che questo è un sistema come un altro per portare il valore che segue la «STA» della locazione $7016 a $0427 che corrisponde alla locazione di memoria del carattere di schermo dell'angolo in alto a destra (il primo carattere che dovrà essere sostituito). Le successive quattro istruzioni ■ partire dalla locazione $700A depositano il valore del puntatore alla mappa, nelle locazioni $FB ■ $FC. Segue la routine che trasferisce 17 dati consecutivi, in 17 locazioni, ognuna distante 40 locazioni dalla precedente. I 17 dati consecutivi sono ovviamente quelli della mappa, mentre le locazioni

Il digitalizzatore per Amiga

Questo riquadro sarà di aiuto a tutti coloro che vogliono adattare il digitalizzatore audio pubblicato in questa rubrica sul numero di ottobre al loro Amiga. È sufficiente munirsi di un cavetto a 10 poli e di un connettore adatto alla porta parallela (25 pin). Collegate i piedini del connettore indicati in foto, con i dieci poli del cavetto e, dall'altro lato, collegate i poli con i pin dell'integrato (ADC-0802). Il polo «14» va collegato all'alimentazione del circuito e il polo «20» alla massa. A questo punto la modifica è completa e il circuito può essere utilizzato unitamente ai programmi di digitalizzazione più comuni, come l'Aegis Audiomaster.





distanziate sono quelle dello schermo caratteri. Proprio a causa di questa differenza di «distanze» tra le locazioni, occorre usare l'artificio della modifica dell'indirizzo di memoria puntato dall'istruzione «STA» visto prima; per saltare 40 locazioni alla volta andiamo a modificare direttamente la RAM che contiene il programma piuttosto che utilizzare un registro indice, quindi è per questo che inizialmente vengono impostate al valore della locazione iniziale. Dopo 17 cicli si esce da questa routine ■ si «resetta» la locazione $D016 quindi si ritorna a $710B per un secondo «reset» che a dire il vero non serve (o meglio non serviva il primo, ma non capisco perché l'avevo messo... probabilmente ho dimenticato di toglierlo... voi potete farlo, se volete, con delle NOP) e quindi si salta a $EA31. Fine dello scrolling.

Megaposta

Caro ingegnere Pesce (cane) Marco, il gioco che stai cercando di realizzare sta venendo abbastanza bene, grazie soprattutto all'opzione per i due giocatori.

Le schermate sono valide, e spero saranno supportate da un adeguato sonoro.

OK, non ti ho scritto solo per complimentarmi, ma per offrirti la mia collaborazione. Praticamente, se hai bisogno di programmatori in L/M, sarei disponibile.

Fammi sapere se ti posso essere utile; in ogni caso auguri per il gioco, che, come fanatico di Elite, non posso fare a meno di approvare.
Conan of CCS alias M. Bottura, Bologna

Caro Conan, il tuo aiuto è indispensabile come quello di molti altri, quindi, approfittando del fatto che adesso non devi più aspettare dei mesi per ricevere ordini, prova a telefonarmi che ne parliamo insieme.

Riprendiamo la lettera di Giorgio Zanella di Poggibosi (SI), che trattava di un argomento riguardante il digitalizzatore di suoni del numero di ottobre. Caro Giorgio, come ti avevo accennato, la soluzione del tuo problema è nel programma «Digital Voice» che apparve nelle pagine dedicate al Super Software del numero 65 (luglio/agosto 1987) di MC. Si trattava di un programma che «catturava» i suoni provenienti dal registratore Commodore ■ li memorizzava in RAM. I suoni potevano essere riascoltati e salvati dunque su disco o cassetta, per poi essere riutilizzati e inseriti nei nostri programmi. Non effettuava alcuna compressione di dati perché, in pratica (a meno di una digitalizzazione con ingresso nullo), il segnale sonoro è troppo complesso per essere adatto a questo tipo di manipolazione. Non capisco però quando parli di nibble... guarda che il segnale del registratore è identificato da un solo bit e quindi non è necessario utilizzare un nibble intero. Il digitalizzatore hardware del numero di ottobre è a otto bit, anche se ne vengono utilizzati solo quattro (ma con l'Amiga invece li utilizziamo tutti... vedi riquadro dedicato), per questo occorre un nibble intero. Nel tuo caso dovresti dividere ogni byte in otto parti e infilarci dentro ■ bit... Quando mi parli di parole memorizzate però non capisco come sia possibile che con il tuo presunto spreco di memoria sia possibile memorizzarne addirittura 5! I casi sono due: o ti sei spiegato male (quanto sono lunghe le parole?) ■ qualcosa non torna. comunque a me sembra che 5 parole siano molte per 64k di RAM... tanto per curiosità, cosa te ne fai dei suoni digitalizzati?

Sono arrivate oggi altre lettere ma non ho il tempo per esaminarle, quindi comunico solo i mittenti, per evitare preoccupazioni: Gianni Gregoroni (FI), Gianmaria Mancosu (Cagliari), Emiliano D'Onofrio (Roma), Paolo Galimberti (Milano).

A proposito dell'ADC-0802; io l'ho reperito anche in un negozio sito nei pressi della mia abitazione... è mai possibile che sia così introvabile?

di Marco Pesce

Un videogioco tutto nostro

«Come sta andando il Megagame? — Abbastanza bene, devo dire; i lettori scrivono e propongono le loro idee, io le esamino e cerco di inserirle... — Non ci sono problemi allora!? — Beh, a volte si sente la necessità di avere un servizio postale più rapido, magari un servizio "telepatico"... — Comunque tutto procede per il meglio mi pare... — Quasi... il più delle volte le scelte vengono fatte senza possibilità di avere un immediato responso degli altri collaboratori e ciò si traduce in "ti piace o non ti piace... te lo tieni lo stesso"... — Ti sembra giusto tutto questo? — Ad essere sincero una soluzione ci sarebbe... — E cosa aspetti a tirarla fuori? — Il problema è che non so quanto sia possibile realizzarla... — E tu prova... semmai ci ripensi, tanto non è il primo ripensamento che fai, no!? — Hai ragione, quasi quasi la dico — Non avere paura, al massimo ti rovini! —»

Bene, quello che avete appena «ascoltato» è il mio discorso fatto con la «voce della coscienza» in questi ultimi giorni. Lo so che siete tutti lì in attesa della «soluzione» e quindi non vi faccio aspettare oltre. Squillino le trombe, abbiamo la nostra linea telefonica diretta con Marco Pesce! Come sarebbe «Cosa vuol dire»!?! Vuol dire che se fino a ieri siete stati a combattere con il famoso ritardo dei «tre mesi», da oggi in poi dovrete combattere con il meno famoso ritardo dei «cinque secondi»... adesso non esageriamo... non vorrei che qualcuno avesse frainteso e si fosse precipitato al telefono nella speranza di accaparrarsi la linea telefonica per primo; abbiamo degli orari ben precisi. Dunque è possibile parlare con il sottoscritto il venerdì, dalle 15.00 alle 17.00... ■ basta! Chiunque tenta di fare il furbo si ritrova con Luana che dice «mi dispiace ma non è in ufficio». E state sicuri che non sta scherzando... Cerchiamo di dirlo in un atteggiamento più serio. Allora, NON abbiamo scherzato; è veramente possibile avere il nostro scambio di informazioni ad alta velocità. Elencarvi i vantaggi che questo «piccolo cambiamento» comporterà al procedere del Megagame mi sembra del tutto superfluo. Il numero lo conoscete; cercate di usarlo nel migliore dei modi e, ancora, non telefonate fuori orario; sprecherete solo tempo e denaro! E per quelli che non si sentono all'altezza di telefonare? Non ne vedo il motivo, comunque la Megaposta continuerà a fare il suo dovere... soprattutto per quanto riguarda l'invio di supporti cartacei o magnetici. Detto questo non ci resta che tornare al nostro Megagame e revisionarlo in funzione delle nuove possibilità di sviluppo. Siamo sicuri che la struttura di gioco attuale lo rende divertente? Non c'è dubbio che un giudizio del genere deve per forza di cose provenire dal parere dei più «critici», per evitare di farsi influenzare dal gusto personale. Ora ne abbiamo la possibilità, quindi sfruttiamola. In ogni modo facciamo alcune considerazioni. Il nostro gioco fa parte di quel genere che mette a confronto due giocatori umani, quindi tutta la serie di giochi sportivi (calcio, tennis, olimpiadi, ecc.) o comunque di giochi strategici tradizionali (scacchi, dama, ecc.). È pur vero tuttavia che anche questo genere di giochi possiede l'opzione di «sfida al computer», ma il più delle volte ciò si rivela essere un ripiego alla mancanza di altri giocatori umani. Qualcuno potrebbe far notare che sfidare il computer equivale ad allenarsi contro un campione per poi battere gli amici e anche questo è vero, soprattutto in un gioco come gli scacchi. Ma cosa c'è dietro questo genere di sfida? In realtà noi stiamo giocando contro il programmatore del gioco che, in generale, avendolo concepito lui, sa quali sono le tecniche vincenti e quindi gioca nel migliore dei modi. Il discorso è un po' diverso se il gioco offre una sfida ■ problemi da sempre rimasti irrisolti (chi conosce la tecnica vincente in assoluto del gioco degli scacchi si faccia avanti) e quindi anche il programmatore ripiega su un algoritmo che ritiene più ■ meno valido, ma non imbattibile. Il massimo sarebbe poter elaborare un algoritmo che si sviluppa sugli errori commessi e ciò, in giochi molto semplici, è possibile. Tuttavia rimane il problema che se il gioco ha una tecnica vincente in assoluto e il programmatore riesce ad elaborarla (non sempre ci riesce, a volte volutamente), non c'è gusto a sfidare la macchina. Essere coscienti di questa situazione e realizzare un algoritmo più semplice per ovviare a quest'ultimo inconveniente è quanto meno una presa in giro nei confronti del giocatore. E il nostro Megagame quante tecniche di vittoria possiede? Non le conosciamo perché abbiamo stabilito delle regole senza ponderare troppo sulla loro influenza effettiva nei confronti dell'esito del gioco. A mio avviso studiare alla perfezione tutte le varie possibilità strategiche a tavolino è un'operazione alquanto noiosa e il tutto si potrebbe risolvere eliminando la possibilità di giocare contro il computer e quindi il dovere di sviluppare un «algoritmo» efficiente (decisione tra l'altro già presa da me in precedenza senza troppi scrupoli). Siete d'accordo su questo punto? O avete voglia di realizzare l'algoritmo? D'altra parte, dando una pur veloce occhiata alla struttura del gioco, difficilmente si potrebbe concludere che non offre sfida, con tutte quelle variabili da modificare! Quale saranno i punti di forza del Megagame? Non c'è dubbio che in ogni gioco il principale punto di

forza è la grafica e se vogliamo uno degli altri è il sonoro. Noi ci siamo impegnati nel realizzare un gioco che offra una sfida sempre nuova perché piuttosto complesso ed è proprio questa complessità che rischia di trasformarlo in un cattivo gioco. Sarà pure una mia convinzione, ma se mi devo scervellare su un problema molto grande, il videogame mi ■ trasforma in una specie di lavoro forzato (...ma chi me lo fa fare?). Nel nostro caso questo ■ meno probabile che ■ verifichi, perché i problemi che incontra un giocatore li dovrebbe incontrare anche il relativo sfidante ■ la frustrazione diminuirebbe senz'altro (un altro punto ■ favore della sfida «faccia a faccia»). L'importante è che le difficoltà non vengano generate dal sistema di controllo, quindi dalla giocabilità. E proprio questo il punto: il Megagame è giocabile? Senza dubbio difficile a dirsi a priori; bisognerebbe testare la gestione dei menu. Possiamo renderla più funzionale? È possibile semplificare la gestione dei dati senza alterare troppo le possibilità strategiche? Credo proprio di sì ■ questo dovrebbe venir fuori da considerazioni fatte a viva voce, proprio utilizzando il nostro nuovo mezzo di comunicazione. Forza ragazzi!

Scuola di videogame

In questa puntata lasceremo un po' da parte il nostro Psicoman per dedicarci alla descrizione di un programma inviatomi da un lettore di Gorgonzola (Milano), Paolo Galimberti. Nel numero di ottobre ho pubblicato la sua lettera nella quale mi diceva che aveva realizzato un programma che permetteva di disegnare aree grafiche per i nostri videogiochi con molta semplicità ■ soprattutto con un ridotto spreco di memoria. Oggi il programma è giunto nelle mie mani e deve assolutamente arrivare anche nelle vostre, sempre che il genere vi interessi.

Parliamo del programma. Una descrizione un po' più precisa ci dice che, partendo da un set di caratteri ridefinito (possiamo ridefinirlo anche con il programma stesso), è possibile realizzare un fondale di dimensioni enormi per il nostro «Antiriad» fatto in casa. In sostanza il programma realizza un file che integrato nel nostro programma-gioco avrà risolto il problema della grafica di sfondo, perché, tra l'altro, sarà possibile, pokando in alcune locazioni, far «scrollare» lo sfondo nelle quattro direzioni. Intuisco già che alcuni di voi stanno pensando ad uno scrolling fluido... Lo pensavo anch'io prima di aver terminato i miei esperimenti di prova, ma devo dire di essere rimasto un po' deluso; lo scrolling è del tipo «a locazioni», il che si traduce nell'impossibilità di adattarlo a giochi stile Ghosts'n Goblins (... appunto parlavo di Antiriad). Se non siete già corsi a piangere dalla disperazione continuate pure a leggere perché, a parte questo «piccolo» neo, il programma è veramente ben realizzato. Abbiamo parlato di ridotti consumi di memoria e di vastissime aree di gioco... ci deve essere qualche trucco. E infatti il trucco c'è ed è anche tra i più conosciuti, grazie ■ quel famoso programma che si chiama S.E.U.C.K. Quelli di voi che si sono cimentati con questo editor di giochi a scorrimento verticale sanno di cosa sto parlando; ■ costruiscono delle «tessere» di 5x5 caratteri che poi serviranno, ■ loro volta, per costruire una «mappa». È chiaro che memorizzare con un solo byte un gruppo di 5x5 caratteri equivale ■ ridurre di 25 il consumo di memoria. In altre parole, con 1000 byte di occupazione memorizziamo 25.000 caratteri o se vogliamo 25 «stanze» del gioco. Inseriamo il dischetto ■ carichiamo... siamo di fronte al menu principale (*foto 1*), joystick in porta 2. Scegliendo «edit blocks» possiamo passare alla costruzione delle «tessere» o «blocks». Ci troviamo subito di fronte ad una richiesta; quale carattere scegliere per gli spazi vuoti? Tutti sappiamo che il valore di default è il 32 ■ anche il programma lo sa... se non abbiamo pretese particolari possiamo lasciare il valore originale e proseguire, premendo il FIRE o il tasto RETURN, oppure cambiare il valore con la tastiera, con il joystick ■ con i tasti cursore, +/−. Con la tastiera funziona anche il tasto DELete. A questo punto, poco meno della metà superiore dello schermo è occupata da una parte dell'area grafica (*foto 2*). Sono visibili sotto i 256 caratteri ridefiniti di default e sulla destra il blocco in fase di «edit». Infine, nella parte inferiore ritroviamo i comandi per cambiare i colori dello schermo, dello sfondo, del blocco, ecc., e per scegliere quale blocco modificare. Una caratteristica molto importante del programma è quella che, se si modifica il colore dello sfondo, l'operazione viene resa visivamente solo nelle aree di disegno e quindi non intacca i colori del menu, rendendoceli magari illeggibili in caso di scelte cromatiche molto sottili (ad esempio: giallo su bianco). Ogni blocco possiede un particolare colore che viene utilizzato per tutti i suoi 5x5 caratteri. L'accesso ai menzionati comandi si realizza con lo spostamento del cursore nella zona di schermo relativa. Possiamo accelerare la variazione premendo, contemporaneamente allo

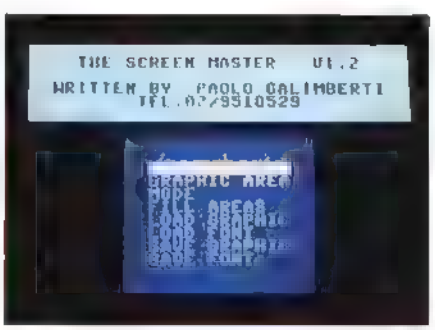


Foto 1

Foto 2

Foto 3

Foto 4

spostamento, il tasto FIRE. Ritorniamo all'area dei caratteri e selezioniamo un carattere, sempre con il FIRE e ci ritroviamo sulla zona del blocco; ancora FIRE ■ il nostro primo carattere è stato inserito. Potete notare che il set di caratteri di default è sì ridefinito, ma non molto adatto per disegnare aree di gioco. Non ci sono chiaramente problemi in tal senso; basta premere la barra di spazio quando siamo con il cursore sul set di caratteri e entreremo nel modo «editor di caratteri». Da notare che è anche possibile caricare un nostro font, magari realizzato in precedenza con un altro editor. Siamo di fronte ad una nuova schermata (*foto 3*). È visibile il carattere grafico ingrandito che abbiamo puntato con il cursore prima di passare a questo schermo. Il numero di opzioni disponibili in questa sezione è notevole e non ci dilungheremo nella loro descrizione, ma del resto molte sono abbastanza intuibili. Una volta preparati caratteri e «tessere» è il caso di passare al disegno della mappa. Selezioniamo «graphic area» e lo schermo mostra una visione della mappa (*foto 4*). Premendo la barra spazio si accede alle opzioni visibili nella parte bassa dello schermo; sono quelle che ci permettono di scegliere le dimensioni della mappa in X ■ in Y. Torniamo alla mappa (barra spazio) e depositiamo i nostri blocchi. Possiamo muoverci per tutta la mappa semplicemente portandoci con il cursore lungo i bordi ■ continuando il nostro spostamento, con scrolling dello schermo. Torniamo al menu principale. L'opzione «fill areas» è molto utile in caso di grosse cancellazioni e ci permette appunto di riempire un blocco, l'area colore o l'area grafica, con un determinato valore (non necessariamente lo «zero»).

Le opzioni di «load» ■ «save» permettono anche di visualizzare la directory (premendo il famoso «$»). Un eventuale font disegnato con un editor esterno deve essere un file programma, ma non importa la sua locazione originaria; il programma sa dove dovrà caricarlo. La «save graphic» serve invece per salvare i restanti dati elaborati dal programma, tranne, ovviamente, il font di caratteri. Terminata la nostra fatica questo file sarà quello da utilizzare unitamente al programma «Compressor», presente sullo stesso dischetto. Detto programma è quello che (... cosa vi dice il nome?) serve ad eliminare le parti superflue, quindi ad ottenere un file finale contenente tutte le routine necessarie per visualizzare il lavoro all'interno dei vostri programmi. Dopo numerose ma necessarie domande il programma salverà su disco il file ■ ci fornirà una scheda «tecnica» della nostra routine. Detta scheda comprenderà l'allocazione della routine, dello schermo video, del set di caratteri ecc. ■ in più ci dirà le SYS e le POKE per farla funzionare. In ogni caso per farla partire occorre una SYS 31744, mentre per disegnare lo schermo una SYS 32464. Per far scrollare lo schermo basta «pokare» in 32459 un valore compreso tra 1 e 4, rispettivamente per andare in alto, in basso, ■ sinistra ■ ■ destra. Se ad uno scrolling di un solo blocco ne preferiamo uno più «massiccio» basterà cambiare il valore della locazione 32460. Tiriamo delle conclusioni. Il programma come detto è ben realizzato e faciliterà notevolmente il gravoso compito di disegnare i fondali, ma la limitazione principale (... solo scrolling «grosso») lo rende utilizzabile solo per giochi particolari, quindi se volete realizzare il vostro «spara ■ fuggi» con scorrimento ad otto direzioni dovete rivolgervi ad altri kit di costruzione, sempre che siate in grado di reperirli.

Megaposta

«Carissimo Marco Pesce, vivissimi complimenti per una rubrica sempre troppo breve, ma che purtroppo soffre di un fastidioso ritardo con le esigenze del Megagame (rit. rivista+rit. posta +rit. scrittura articolo) ma d'altronde non puoi farci niente. In qualunque caso sei un disgraziato ■ passo subito ■ dirti il perché. Il tuo circuito campionatore, basato sul convertitore analogico-digitale ADC 0802, è il punto di partenza per software di campionamento della potenza non trascurabile e di facile realizzazione. Ed è proprio in questi sogni che mi crogiolavo, mentre mi accingevo alla realizzazione, quando mi sono accorto di un piccolissimo particolare: nei negozi di componenti elettronici di Cagliari manca completamente il preziosissimo ADC 0802!... ...tanti saluti ■ grazie ancora per lo splendido progetto che non potrò mai realizzare».

Gianmaria Mancosu, Cagliari

Accidenti Gianmaria, sai che non sei mica l'unico ad avere questo problema? Di chi ■ la colpa? Della distribuzione di componentistica elettronica? Cosa possiamo farci? Comunque molti negozi ultimamente ■ stanno stranamente rifornendo di «preziosissimi ADC 0802» ...chissà che anche a Cagliari non succeda qualcosa di nuovo. Per quanto riguarda il ritardo «generale», non so se hai già letto l'articolo di questo mese (molti cominciano dalla Megaposta), ma un tentativo per eliminarlo lo abbiamo fatto. ■■

di Marco Pesce

Un videogioco tutto nostro

Comincio a tremare...
A tremare per quello che (per me) dovrà accadere tra non molto (... le vostre telefonate)
Speriamo che ne venga fuori qualcosa di buono... E nel frattempo? Facciamo una bella analisi di coscienza (sì, ancora una)!

Ormai dovrete essere abituati ai cambiamenti ed è per questo che non mi preoccupo più di tanto nel comunicarvi che sono riuscito ■ semplificare l'interfaccia (menu) giocatore/pianeti (come foto dimostrano). Ciò vuol dire che gran parte delle considerazioni fatte in precedenza vanno a farsi benedire, anche se sono indubbiamente state utili per l'impostazione generale del gioco. Passo subito al dunque. Quello che non mi andava giù dell'interfaccia precedente era la macchinosità di una qualunque operazione; ora, come potrete notare dai dettagli seguenti, le cose si sono semplificate ■ la densità delle opzioni «tutte in schermo» è aumentata, il che permette di avere un confronto più immediato tra le variabili.

E passiamo alle modifiche fatte. Una delle più evidenti è la riduzione del numero di pianeti presenti nella mappa. Non solo sono scesi di numero, ma sono anche diventati più omogenei; non esistono più pianeti che non siano abitabili! Il perché di questa e di tutte le modifiche che troverete ha una sola risposta: semplificare ciò che non è di importanza rilevante. Di pianeti non utilizzabili ce ne saranno anche, ma sulla mappa ci sono solo quelli abitabili e quindi da conquistare.

Passiamo con lo sguardo sulla foto 1. Il passaggio da una sezione ad un'altra dei menu si è notevolmente velocizzato; basta pigiare su uno dei 4 pulsanti in alto a sinistra (sempre presenti). Nella fotp 1 è in evidenza l'opzione per agire sulle strutture del pianeta in considerazione attualmente (scelto con l'opzione MAPPA). Quello che prima era gestito da più schermate ora è concentrato in metà schermo. Possiamo costruire, distruggere, disattivare e riattivare strutture, sia di ricerca che di produzione, tutto questo con la semplice pressione di alcuni pulsanti. E in più possiamo avere (sulla destra) un resoconto dello stato generale del pianeta, sempre utile. Da notare che le varie grandezze non sono più visualizzate sotto forma di numeri, ma con degli indicatori, certamenti più semplici da identificare e da manipolare. Chiedetevi pure se ciò ha comportato modifiche alla struttura dati dei pianeti... Cerrrrto!!!

Praticamente tutte le grandezze entrano in un solo byte e la maggior parte di esse non lo utilizzano nemmeno per intero. Riepilogo della struttura dati di un pianeta:

— *posizione nella mappa (16 in x,16 in y);*
— *superficie disponibile;*
— *superficie utilizzata;*
— *abitanti;*
— *possessore attuale;*
— *installazioni produttive (attive e non attive in due byte distinti):*
1) energia
2) alimenti
3) minerali
4) armi
— *centri ricerca (attivi e non attivi in due byte distinti):*
1) energia
2) alimenti
3) minerali
4) armi
5) trasporti
6) medicine
— *risorse:*
1) energia
2) alimenti
3) minerali
4) armi
— *numero tipi di risorse: (due byte ognuna)*
1) energia
2) alimenti
3) minerali
— *stato attuale (guerra o pace)*

e abbiamo finito con la struttura di un pianeta. C'è poi la struttura che indica lo stato globale della civiltà (visualizzabile in qualunque momento con il tasto «GLOBALE»). Essa contiene:

— *energie;*
— *alimenti;*
— *minerali;*
— *potenza armi;*
— *potenza medicine;*
— *viaggi (velocità ■ costo);*

il «numero tipi di risorse» occupa due byte perché non è memorizzato come semplice cifra, ma ogni bit «acceso» corrisponde ad un particolare elemento in più, quindi al max possono essere 16.

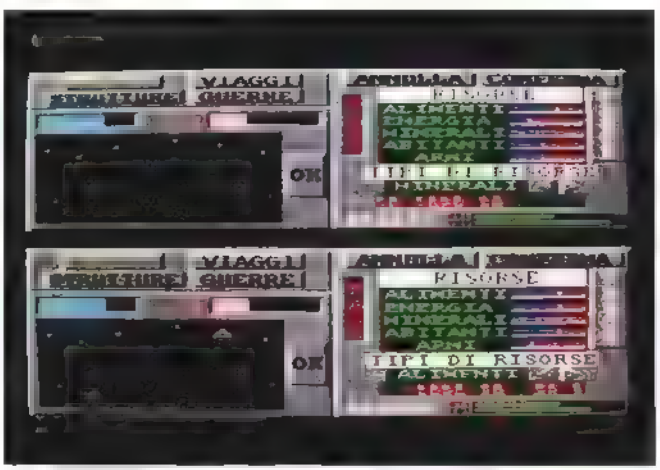
Occorre una ulteriore quantità di byte nella struttura di ogni pianeta per indicare in modo «normale» quanti sono i «tipi di risorse» (serve per velocizzare i

Foto 1

calcoli delle routine) e per indicare quali sono i possibili elementi che si potranno «scoprire» su quel pianeta (questa informazione ovviamente è «segreta» e solo il C64 la conosce, per i suoi calcoli).

Torniamo agli «indicatori»; essendo questi formati da pixel non possono essere di tipo analogico e quindi posseggono un limitato numero di posizioni, che dipende ovviamente dalla loro grandezza. Ciò si traduce nel fatto che è inutile memorizzare una grandezza con possibili oscillazioni da 0 a 65.000 se poi di ciò non ce ne possiamo rendere conto. E poi, diciamo la verità, non serviva tutto quel range di possibili valori: «viva la semplificazione»! In un byte possiamo far entrare anche un indicatore di 255 pixel che comunque sullo schermo non c'entra, quindi ne useremo, ad esempio, 50. Il byte relativo allo spazio disponibile viene usato per intero anche se sullo schermo ne viene visualizzata una sintesi. Il motivo di detta eccezione è presto detto: non è molto bello vedere le nostre barrette sempre verso il minimo valore solo perché non c'è più spazio. Mi sono accorto che non potete comprendere quanto appena detto se non siete a conoscenza di una ulteriore modifica fatta alle unità di misura e ai consumi in generale. Seguitemi. Tutto è stato reso uniforme ed ora un pixel (unità di misura dettata dalla natura degli indicatori) di abitanti occupa quanto un pixel di energia, o quanto uno di armi, o quanto uno di installazioni, ecc., ecc. Quindi se avessimo fissato lo spazio disponibile a 50 pixel come valore max (ricordiamoci che ogni pianeta ha una superficie diversa da un altro), ci saremmo trovati nelle condizioni di esaurire lo spazio in corrispondenza del max valore di abitanti (come sarebbe giusto). In realtà su un pianeta non possono trovarsi solo abitanti (poveretti) quindi il max valore dell'indicatore non sarebbe mai stato raggiunto in nessun caso. Se invece, come abbiamo fatto, mettiamo a disposizione della superficie 255 pixel virtuali (sempre come max valore), una speranza di vedere giungere la nostra barretta a fondo scala la avremo! Riprendiamo l'«interessantissimo» discorso sui consumi e sui prodotti. Ogni centro (di qualunque tipo) consuma un pixel di energia e uno di minerali «al giorno» e, nel caso dei centri produttivi, produce un pixel della sua specialità moltiplicato per il «numero di tipi» disponibile sul pianeta. Mi sembra evidente che se inizialmente un pianeta possiede solo un «tipo» non avremo alcun rendimento da quelle installazioni (anzi avremo delle perdite).

Foto 2



Unico consumo fuori norma è quello relativo alla «costruzione» di una installazione, che richiede 4 pixel, sia di energia che di minerali, ovviamente. Il consumo degli abitanti è relativo solo agli alimenti ed ogni pixel di abitanti ne consuma... un pixel.

Finita questa tremenda rivelazione di modifica, che manda all'aria tutte le routine che avevamo progettato di realizzare per il calcolo della vita di un pianeta (non disperate, solo un lettore, Giulio Ceriola di Bari, finora mi ha mandato il suo lavoro e troverò ugualmente il modo di ricompensare il suo sforzo) torniamo a descrivere i nuovi menu.

Il riquadro sulla destra è praticamente visualizzato per ogni opzione e si riferisce sempre al pianeta attualmente considerato, quindi, nel caso di opzione che coinvolge due pianeti, al pianeta che stiamo trattando (ad esempio, SORGente o DESTinazione). Sulla destra troviamo anche l'indicatore di PACE o GUERRA e due pulsanti per confermare o cancellare quanto abbiamo modificato con opzione trattata al momento. Per i tipi di risorse ho utilizzato un sistema di visualizzazione un po' più compatto; occorre scegliere con i tasti-freccia la risorsa che ci interessa e l'indicatore a LED si accenderà nel modo più opportuno. Anche per selezionare il tipo di costruzione ho adottato lo stesso sitema.

Passiamo alla foto 2. La mappa contiene un numero inferiore di pianeti, come detto (16 dovrebbero ugualmente

Foto 3

Foto 4

bastare), ■ il sistema per selezionare quello che ci interessa è decisamente migliorato; innanzi tutto abbiamo subito una indicazione del suo contenuto con il pannello di destra e poi sulla mappa sono indicati i pianeti che si trovano in pericolo (cioè da soccorrere) con un triangolino. Come al solito è possibile visualizzare solo «i nostri», solo «i suoi» o «tutti quanti» (pigiando sul LED opportuno). In più abbiamo anche un indicatore relativo alle due quantità. Accortezza molto importante è quella relativa al fatto che, in qualunque opzione ci troviamo, possiamo passare alla mappa e scegliere un nuovo pianeta e poi ritornare all'opzione, senza dover ricominciare da zero con le modifiche. Mi spiego con un esempio; se stiamo tentando di attaccare un pianeta (pannello di foto 4) e ci accorgiamo che le risorse del pianeta attaccante non sono sufficienti, basta passare alla mappa ■ cercare un pianeta più fornito, quindi tornare al pannello delle guerre e accorgersi che non dobbiamo andarci ■ ricercare quale era il pianeta da attaccare, perché non c'è stato un reset delle scelte precedenti (in questo caso, del pianeta da attaccare)... a me sembra comodo. In definitiva quindi, il pianeta (o i pianeti) che stiamo trattando resta valido anche per il resto delle opzioni, fino a quando non viene cambiato con la mappa.

Foto 3 ovvero il pannello dei viaggi. Mi sembra che la foto sia abbastanza esplicativa; selezioniamo i due pianeti con la mappa (abbiamo anche la possibilità di SWAP tra pianeta SORGente e DESTinazione; vedi il piccolo gadget situato tra i due pulsanti) ■ scegliamo la quantità di «roba» da trasportare. Confermiamo e il nostro materiale sarà già in viaggio. Notiamo che ■ fondo a sinistra ci sono altri due pulsanti che permettono di scegliere se iniziare un nuovo viaggio o visualizzare lo stato di uno vecchio; la prima metà dello schermo (pannello del primo giocatore) mostra la scelta «nuovo», mentre la seconda mostra quella opposta. In quest'ultimo caso possiamo soltanto controllare il tempo che resta al termine del viaggio, quindi il materiale che viene trasportato e i due pianeti coinvolti. La visuale di un particolare viaggio si effettua scegliendolo con le solite freccine, tra quelli attivi. Il pannello di riepilogo di destra mostra la situazione di uno dei due pianeti (quello selezionato con il pulsante).

Ultima foto: le guerre. Anche qui non ci sono grosse incognite da svelare. Basta selezionare i due pianeti, quindi la quantità di armi-abitanti da inviare e... attenti ai botti! Ovviamente, possiamo scegliere se iniziare un nuovo combattimento o visualizzare lo stato di uno vecchio, quindi decidere se è il caso di ritirarci, solo se è un nostro attacco (sempre per la solita questione del pianeta che non si abbandona fino all'ultimo sangue).

Come avrete notato, non ho parlato della struttura dati della nostra astronave, né di come è possibile controllarla ■ questo perché non sono più tanto convinto della bontà di inserire la sezione in tre-d o comunque una sezione di SEF (Spara e Fuggi). La vedo un po' come una parte che fa perdere fascino al gioco; perché mai un comandante dovrebbe mettersi alla guida di UNA navicella quando ce ne sono MILIONI da distruggere (il classico «assurdo» di questo tipo di giochi)? E poi, chi ha bisogno di un altro SEF? È vero che di tacito accordo avevamo deciso di inserirlo per ravvivare il gioco con un po' di azione, ma, prima di fare spropositi, ne vale veramente la pena? O conviene realizzare qualcosa di più impegnativo (chi ha detto che deve per forza essere noioso), che coinvolga in modo più adeguato il nostro capo supremo e la nostra testolina? Devo confessarvi che al momento non ho ancora in mente una possibile soluzione al problema. Speriamo che venga fuori dalle nostre chiacchierate!

Scuola di videogame

Tratteremo, in questa e nella prossima puntata, un argomento che dovrebbe essere tra i più interessanti per gli appassionati costruttori di giochi alla PSICOMAN e cioè del come è impostata una routine per la gestione dei movimenti del giocatore

Il test della posizione del joystick ci permette di decidere quale tipo di con-

trollo dovremo effettuare e quindi quale routine dovremo eseguire in questa «passata di RASTER». Se il joystick è impostato verso sinistra il nostro giocatore-sprite può trovarsi nelle seguenti condizioni:
1) al limite sinistro dello schermo;
2) in caduta;
3) in salto;
4) iniziare una caduta;
5) iniziare un salto;
la prima di queste condizioni può interessare solo il caso, appunto, di joystick rivolto verso sinistra, mentre le altre sono abbastanza generiche. Fatto sta che se si verifica la prima condizione questa direzione del joy non comporterà alcuna azione e quindi salteremo il pezzo che ci dovrebbe decrementare la coox (coordinata x). Le condizioni 4 e 5 si possono verificare solo in particolari situazioni, che esamineremo tra breve. In particolare la condizione 5 può verificarsi solo se, unitamente ad altre condizioni, il joy è rivolto anche verso l'alto. Per il momento basta controllare se ci troviamo o meno al limite sinistro e quindi se dobbiamo ■ meno decrementare la coox dello sprite. Un secondo controllo, da effettuare ovviamente solo se il primo non ha avuto successo, è quello della direzione destra. In tal caso possiamo trovarci nella condizione di limite destro, che non corrisponde alla max coox dello sprite ma al centro dello schermo, oppure nella condizione di non-limite. La prima ci dice che dovremo effettuare uno scrolling dello schermo, mentre la seconda dice che dobbiamo incrementare la coox. Effettuati questi primi due test possiamo passare al controllo del salto. Il salto può essere nuovo o vecchio ■ cioè ne possiamo cominciare uno in questo momento o dobbiamo continuarne uno iniziato in una precedente passata. Il salto nuovo può verificarsi solo se si abbinano le tre condizioni seguenti:
1) il joy è rivolto (anche o solo) verso l'alto;
2) il giocatore non sta già saltando;
3) il giocatore non sta cadendo;
se non si verifica anche una sola di queste condizioni il nuovo salto non può essere attivato. Si prosegue quindi con un nuovo test.

Se il nuovo salto è possibile lo si attiva (settando un flag) e si passa alla routine di salto.

Tuttavia, nel caso si fosse verificata la condizione opposta alla 2, dobbiamo «saltare» ugualmente alla routine di... salto. Questa routine provvede ad effettuare le seguenti operazioni:
1) decrementare in modo opportuno la cooy;
2) testare l'eventuale presenza di un carattere «solido» sotto i piedi del giocatore;
3) testare l'eventuale fine della fase di salita;
per svolgere la fase 1 occorre fare riferimento ad una opportuna tabella contenete valori via via sempre più piccoli che simulino una salita influenzata dalla forza di gravità. Una possibile tabella è la seguente:
5,5,5,4,4,3,2,1,0;
essa in pratica contiene tutti i valori che dovranno essere sottratti alla cooy, in più passate, prima di cominciare la fase di ricaduta. Da notare che ci occorre anche una puntatore per questa tabella, che in caso di nuovo salto viene «azzerato» alla max velocità (perché in sostanza di questo si tratta).

Quando questi valori terminano (ovvero quando si giunge allo zero) deve cominciare la fase di caduta (operazione 3). Detta fase può cominciare anche in anticipo nel caso il giocatore si trovasse ad aver raggiunto la posizione 0 della cooy. Oppure la fase può essere interrotta dall'eventuale presenza di un carattere solido sotto i piedi del giocatore. Questa condizione è abbastanza delicata ■ necessita di ulteriori dettagli. Innanzi tutto un carattere solido è un particolare carattere, tra quelli del set a nostra disposizione, che deve vincolare il movimento del giocatore, quindi deve impedirgli di cadere, di avanzare ■ di indietreggiare quando gli si presenta rispettivamente, sotto i piedi, davanti e dietro.

Detto questo non rimane che stabilire come effettuare questi controlli. L'operazione è più semplice di quanto potrebbe sembrare. Ed è sufficiente «convertire» le coordinate dello sprite in coordinate «carattere» ovvero determinare quale è il carattere che viene coperto dall'angolo in alto a sinsitra del nostro sprite. Per la coox basta dividerla per 8 ed otteniamo la coox-carattere. Per la cooy il discorso è lo stesso e una volta ottenuta la cooy-carattere dobbiamo sommare tante volte 40 alla coox-carattere quanto ci indica la cooy-carattere, per ottenere infine il carattere effettivo (ovvero il delta da aggiungere alla prima locazione della memoria carattere). Ottenuto questo valore possiamo fare i controlli che vogliamo, semplicemente aggiungendo ad esso un opportuno delta a seconda del carattere che vogliamo testare (ad esempio, per testare il carattere «sotto i piedi», tenuto conto delle dimensioni di uno sprite, basta aggiungere 1 carattere in x e 3 caratteri in y quindi 121 in totale).

Chiusa questa parentesi torniamo al giocatore che supponiamo giunto alla max salita senza aver incontrato alcun carattere solido. In tal caso ci troviamo nelle condizioni di dover attivare la fase di caduta. Ciò si traduce nel semplice settaggio di un flag; la routine seguente si occuperà del resto. Essa è in grado di proseguire una caduta in atto oppure di iniziarne una nuova. E noi la lasciamo nel dubbio per un mese... a risentirci.

Megaposta

Caro Marco, mi chiamo Giulio Ceriola, fortunato possessore di un Amiga 500 e di un C64. Sto seguendo con interesse la rubrica Megagame 64 sin dalla prima puntata, da quando ho letto che doveva essere un gioco alla cui scrittura potevano partecipare tutti.

Così quando ho visto su MC n. 80 che la grande "sfida" (perché penso che questa proprio lo sia) è iniziata, mi sono messo all'opera per scrivere quei programmi da te richiesti.

Ed eccoli qui. Lo so che non sono granché, ma come si dice... tentar non nuoce. Infatti alla lettera allego sei listati di 3 programmi...

Giulio Ceriola, Bari

Ah, se tutti fossero come Giulio... Indubbiamente hai fatto il tuo lavoro, per lo meno, con dedizione e per questo meriti di essere elogiato, anche se probabilmente solo una piccolissima parte delle tue routine sarà effettivamente utilizzata. In ogni caso complimenti e prosegui su questa strada.

... Le (o ti?) scrivo per esporre alcune idee balenatemi nella testolina dopo la lettura dei fantasmagorici number di novembre e gennaio...

Ivan Melle, Torino

D'accordo Ivan, hai detto la tua. Continua con nuovi ■ magari più impegnativi suggerimenti...

di Marco Pesce

Un videogioco tutto nostro

Quanti mesi sono passati dal nostro primo appuntamento? Beh, a questo punto davvero molti... Che cosa si era promesso di fare Marco Pesce nel lontano dicembre del 1987? L'intenzione era quella di organizzare la realizzazione di un grande videogioco, usufruendo della partecipazione dei lettori. Molti accettarono entusiasticamente l'invito e il primo passo fu quello di definire la trama del game

Naturalmente ognuno propose un concetto di gioco a lui congeniale, ma quello che si voleva ottenere era un gioco che oltre ad essere innovativo avesse soddisfatto un po' tutti. Si scartarono quindi tutti i giochi che trattavano argomenti particolari, ma del resto non era possibile trovare una soluzione perfetta, quindi, visti i gusti della maggioranza dei lettori, si convenne di realizzare un war game spaziale. La definizione della sua struttura durò per molti mesi («grazie» alle inevitabili difficoltà di carattere tecnico) e anche al momento attuale non è ancora perfetta, ma per fortuna mancano solo dei piccoli dettagli che verranno esaminati in questa puntata. Questo è lo stato dei fatti.

Veniamo al dunque. Manca da stabilire come avviene l'incremento della popolazione e come esso viene influenzato dalla potenza dei medicinali, quindi manca la quantificazione dei costi e dei tempi per ciò che riguarda i viaggi, come avvengono le scoperte e infine mancano le condizioni iniziali delle due civiltà. Tutto qui. Il mese scorso ci eravamo lasciati con un interrogativo; cosa inserire per compensare la mancanza dello «spara e fuggi». Bene, ora abbiamo la risposta; non inseriamo un bel niente, perché il SEF non ha creato un grande vuoto e il gioco va bene anche così. Mancano la presentazione, il finale e la colonna sonora, che potranno usufruire di tutta la memoria che vogliono, visto che il gioco, per come è impostato finora, non ne occupa molta, a meno che non ci sarà qualcuno tra voi che vuole inserire schermi grafici e animazioni di contorno; a mio avviso non sono indispensabili, ma senza ombra di dubbio, se ben studiati, arricchirebbero di molto l'atmosfera del gioco. A tal proposito vi invito ad inviare i vostri suggerimenti; come al solito scrivete la vostra missiva e speditela qui in redazione. Ma prima di continuare con questi discorsi torniamo a ciò che abbiamo accennato nelle prime righe. Cominciamo con lo stabilire i costi e i tempi dei viaggi. La mappa è composta da una griglia di 16×6 caratteri e in tale area sono piazzati i sedici pianeti abitabili, ciascuno collocato nel proprio sistema solare. Come tutti sapete, i centri di ricerca permettono di scoprire sistemi più rapidi e/o meno costosi per viaggiare; occorre stabilire dei costi e dei tempi validi inizialmente. Il «volo» da una casella (carattere) ad un'altra ci costa un «quanto» di tempo, che poi sarebbe l'unità di misura standard valida per tutti i pianeti (all'incirca equivalente alla durata di un giorno terrestre). Tale «quanto» ci permetterà, con l'evolversi dei mezzi tecnici, di percorrere spazi sempre più ampi. I costi sono funzione della quantità di materiale trasportato (ma non della distanza) e in condizioni iniziali equivalgono a 1 unità di energia per ogni 10 unità di materiale utile. Anche in questo caso, grazie ai centri di ricerca, a parità di costi, il materiale utile trasportabile verrà incrementato. Gli indicatori del pannello «GLOBALE» contengono, per il tempo, la quantità di caselle che possono essere percorse (1) e, per i costi, la quantità di materiale che è possibile trasportare (10). Il tempo necessario si arrotonda sempre all'unità superiore, quindi se per caso con un «quanto» di tempo possiamo percorrere 4 caselle, ma il viaggio in questione ne comprende solo 3, la durata sarà ugualmente di un «quanto». La massima distanza tra due pianeti è di 16 caselle, quindi i tempi saranno ridotti fino a questo limite (un «quanto» per 16 caselle). I costi potranno ridursi fino ad un massimo di 48 unità di materiale per unità di energia.

E passiamo all'incremento della popolazione. Un calcolo spartano ci porterebbe alla conclusione che essa raddoppia di volta in volta e ciò chiaramente è impossibile; possiamo invece stabilire che per una certa percentuale, tanto più bassa quanto più alta è la potenza dei medicinali, la popolazione viene decrementata. Se poniamo a 16 la max potenza dei medicinali e, ad esempio, ■ 4 la potenza attuale diremo che per ogni 16 abitanti 4 devono essere «decre-

mentati». Chiaramente se venisse a mancare il supporto alimentare detto decremento sarebbe di gran lunga maggiore.

Come avvengono le scoperte? Anche in questo caso entra in gioco la probabilità. Per verificare se una scoperta deve avvenire o meno si somma il numero di installazioni attive sul totale dei nostri pianeti, per ogni tipo di ricerca, ■ si esegue una «estrazione» basandosi su un numero random; se questo coincide con il numero di installazioni a nostra disposizione o è comunque inferiore, avviene una scoperta. Il range di escursione di tal numero è da stabilire e ciò verrà fatto dopo la programmazione delle routine. Anche le condizioni iniziali dovranno essere stabilite a routine ultimate, al fine di regolare a puntino la giocabilità; è infatti abbastanza inutile farlo ora se non sappiamo quale sarà l'effetto reale che esse avranno sul gioco. Stabilite queste ultime specifiche la struttura del gioco è praticamente completa, ma non è da escludere la presenza di altri piccoli buchi, che tuttavia verranno, nel caso, eliminati da chi si occuperà della programmazione. A tal proposito bisogna ammettere che una partecipazione, in tal senso, da parte dei lettori è molto difficile da ottenersi ■ probabilmente questo compito spetterà integralmente al sottoscritto; se non siete d'accordo fatevi avanti. Restano ancora irrisolti i seguenti 2 punti:

— Presentazione
— Colonna sonora

e anche questo è compito di voi lettori. Pensate a una qualunque introduzione che possa far colpo sul «potenziale» giocatore e mandatemi uno schema teorico; ricordatevi tuttavia che stiamo lavorando su un C 64!

Per la parte sonora mi servono dei bravi musicisti; se siete bravi anche con il linguaggio macchina cercate di inviarmi una routine funzionante in IRQ, altrimenti mandatemi una musicassetta, o, al limite, aspettate che l'editor presentato nella scuola di videogame in questa puntata sia pronto.

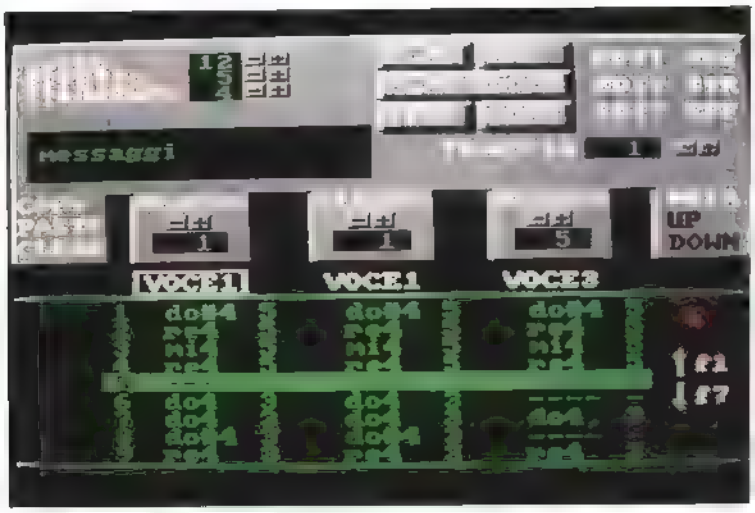
Inventate una storia valida e...

Il Megagame ha bisogno di una storia, una storia che racconti ciò che ha provocato lo «stato attuale delle cose». Tra i nostri lettori ci saranno senz'altro degli ottimi «scrittori di fantascienza» in grado di risolvere il problema. Proprio per questo motivo vi invito ad inviarmi la vostra storia. La migliore tra quelle che giungeranno in redazione sarà inserita nella presentazione del game e pubblicata su MC.

... Anche voi potrete vantarvi di aver partecipato alla realizzazione del Megagame.

Scuola di videogame

Secondo e ultimo appuntamento con la routine di gestione-movimento del nostro Psicoman e primo appuntamento con un programma per la creazione di routine musicali, fatte su misura per essere inserite nei vostri game. Ma concludiamo il discorso della puntata precedente. La routine di caduta è molto simile a quella di salto eccetto il fatto che l'accelerazione questa volta è concorde al verso del giocatore. La tabella accelerazione ha il suo puntatore posizionato sul minimo valore, che poi sarebbe lo zero; il puntatore viene spostato fino alla max velocità e, da quel momento in poi, continua con tale velocità fino ■ quando non incontra uno dei famosi «caratteri solidi» e resetta il modo caduta. La «caduta effettiva» (quella che abbiamo visto era una «ricaduta») avviene quando, in seguito ad uno spostamento in orizzontale, il giocatore non è più vincolato da un carattere solido. Contrariamente a quello che accade in caso di «ricaduta», il puntatore alla tabella delle velocità potrebbe anche non essere posizionato al minimo valore, quindi se siamo in condizioni di attivare una «caduta effettiva» dobbiamo resettarlo. Resta da considerare l'ipotesi di joy rivolto verso il basso. Tale azione ha effetto solo se siamo poggiati sul solito carattere solido e comporta l'abbassamento del nostro personaggio. La routine che esegue il controllo deve solo impostare il giusto frame di animazione, ed è proprio di animazione che parleremo ora. Innanzi tutto esiste un flag che ci dice se il giocatore è rivolto verso destra o verso sinistra. Tale flag viene aggiornato con lo spostamento del joy e ci servirà per impostare i giusti frame. Veniamo quindi ai frame, che sono fissi in caso di salto di caduta ■ di abbassamento, ■ devono variare con una certa velocità e in una certa quantità in caso di «camminata»; Psicoman possiede, per tale animazione, 4 frame per direzione. Quando la ragazza è ferma c'è una piccola finezza: con la paura di essere attaccata da un momento all'altro continua a voltare la testa a destra e a sinistra! È un piccolo accorgimento che si ottiene semplicemente variando lo sprite superiore (vi ricordo che è formata da due sprite sovrapposti) ogni tot cicli di raster. Se siamo in salto o in caduta o fermi o abbassati si settano gli



vogliamo utilizzare l'editor come se si trattasse di un registratore a nastro, basterà impostare anche il tasto play; in tal modo saremo in grado di suonare «dal vivo», mentre il blocco «scorre» a velocità costante sullo schermo, spazio dopo spazio. Una «traccia» è l'insieme di tre blocchi, ciascuno appartenente a una voce del SID. I blocchi (max 255) possono essere utilizzati in più tracce e in più voci. La voce selezionata sarà quella che verrà «incisa» dalla registrazione. Ci sono anche opzioni per copiare blocchi o per cancellarli, quindi per «spostare» di una nota, verso i toni acuti o i toni bassi (up/down), un blocco. Non mancano funzioni come l'INSERT o il DELETE, utili al fine di correggere eventuali errori e indispensabili al fine di non riscrivere completamente un blocco solo perché una nota è andata fuori ritmo. I tasti funzione (indicati nella foto) ci permettono di muoverci nel blocco e quindi, tra l'altro, di iniziare la registrazione in qualunque punto. In caso di rec-play una eventuale incisione precedente produce un «accompagnamento» (molto utile). C'è anche la possibilità di selezionare lo strumento, o l'«effetto» con il quale la nota sarà suonata. Un secondo pannello ci permette di creare i nostri strumenti; oltre ai normali ADSR e filtri vari è possibile impostare delle modulazioni, associate alla frequenza, al taglio dei filtri o anche al duty-cicle, ma di questo e delle restanti possibilità parleremo nella prossima puntata, quando il programma sarà, probabilmente, già disponibile ai lettori.

sprite opportuni aiutandosi con il valore contenuto nel flag «direzione». Resta da esaminare come funziona la routine di sparo. Un flag ci dice se un «colpo» è già attivo oppure se è possibile spararne uno nuovo. Se si verifica quest'ultima condizione ed il tasto fire è premuto il flag viene alterato e le coordinate dello sprite di sparo vengono fatte coincidere con quelle dello sprite superiore della ragazza. Viene inoltre impostata la direzione di movimento. D'ora in poi ad ogni ciclo raster la coox dello sprite di sparo verrà alterata, fino all'esaurimento della «gittata» o fino al rilevamento della collisione con uno dei nemici. A tal punto lo sparo sarà nuovamente disponibile. Parliamo ora di musica...

Al fine di inserire, nei propri videogiochi, sostegni musicali di notevole o comunque buona fattura, è necessario realizzare routine funzionanti in IRQ, che gestiscano il SID nel migliore dei modi. Una routine di questo tipo non è di difficile realizzazione, ma riuscire ad essere creativi senza il supporto di un editor valido è impresa quanto meno ardua oltre che noiosa. Quello che esamineremo in questa e nelle prossime puntate è appunto un editor, che ci permette di scatenare la nostra fantasia musicale senza dover badare troppo a problemi di carattere tecnico. Il prodotto finale sarà una routine, per l'appunto funzionante in IRQ, inseribile in qualunque listato. Il programma si basa su un concetto molto semplice; creare dei «blocchi» di note che, messi nella giusta sequenza, diano origine al nostro brano. Ogni blocco è composto da 64 spazi. In uno spazio può esserci una nota oppure una pausa. Per inserire le note, nel blocco selezionato, occorre «premere» il tasto REC (vedi foto 1), «puntando» il cursore (freccia rossa) controllato dal joy e pigiando il tasto «fire». A questo punto la tastiera del C 64 si trasforma in, una tastiera musicale ed è possibile suonare ed inserire le giuste pause saltando spazi. Se invece

Megaposta

Caro Marco Pesce, ti scrivo con questa stampante «manuale» in quanto quella meccanica ci ha momentaneamente lasciato. Ancora complimenti per il Mega, ho visto le foto sul numero 82, non credevo ai miei digitaliz... ehm ai miei occhi!! Ma in 64K c'entrano davvero? (spero di sì). Ma sono qui per parlare d'altro, l'introvabile ADC-0802 io l'ho trovato sotto casa, però costa più di 32.000 lire, ho quindi utilizzato l'ADC-0803 e devo dire che funziona a dovere, anche i piedini coincidono e digitalizza che è un piacere!... Per chi volesse conoscere il prezzo dell'803 rispondo che non ricordo, mi sembra sulle 16.000 o giù di lì. Qua da Firenze è tutto, salutoni e a presto.

Gianni Gregoroni - Firenze

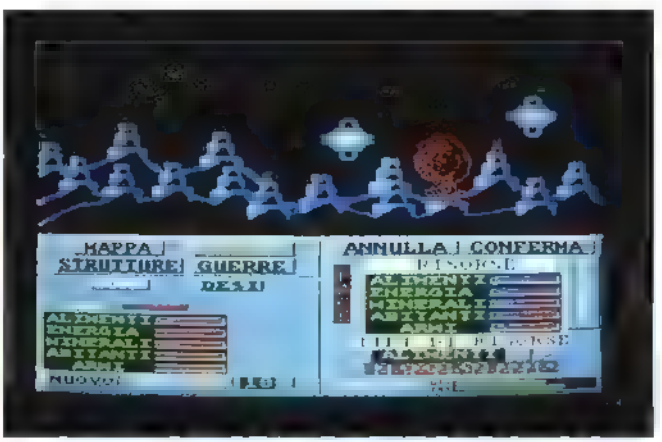
di Marco Pesce

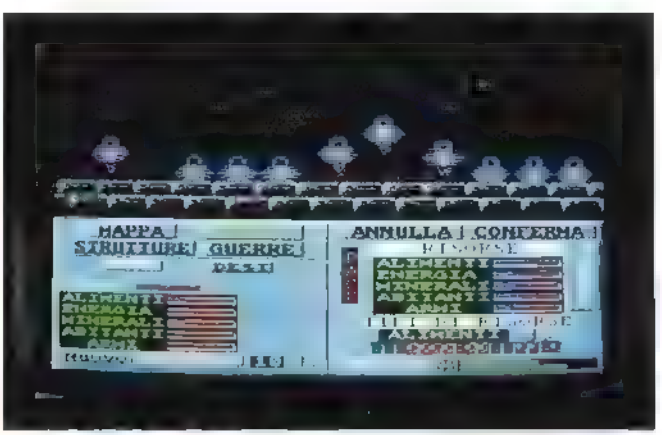
Un videogioco tutto nostro

Conclusa la interminabile massa di elucubrazioni relative alla struttura del Mega non ci resta che dedicarci alla realizzazione del lato estetico, quindi presentazione, musica e grafica animata

Nella scorsa puntata vi ho invitato a farmi pervenire i vostri pareri in merito ■ naturalmente dovrò aspettare ancora qualche giorno prima di riceverli.

Grafica! Che bella parola. Cosa sarebbero i videogiochi senza di essa?... Dipende. Dipende dal tipo di game. Il Megagame abbiamo visto che non ne desidera molta, ma di questi tempi tutti hanno grafica, quindi anche il Mega dovrà averne. Il «problema» e dove inserirla. Le varie operazioni che i menu ci permettono di effettuare potrebbero essere intervallate da scene animate. Per realizzarne un numero considerevole è indispensabile utilizzare un modo grafico poco dispendioso, quindi non l'alta risoluzione; sprite e caratteri ridefiniti in grande quantità dovrebbero fare al caso nostro. Combattimenti, decollo di astronavi, costruzione di installazioni, interni di centri di ricerca, carico di astronavi di trasporto, espressioni di sofferenza o di gioia degli abitanti, distruzione di edifici ecc., queste sono le scene da rappresentare! Possiamo anche servirci dello scrolling, in modo da effettuare «panoramiche» dell'ambiente. Comunque vadano le cose la grafica non deve disturbare l'interazione con il gioco, quindi tutte le scene devono essere eliminabili tramite pressione del testo fire. Quello che mi aspetto da voi sono dei consigli per le «inquadrature» ■ per le ambientazioni, magari supportati da qualche esempio pratico. Diamo un'occhiata alla sezione «audio». Non è escluso che accanto alle previste colonne sonore (due, una per l'introduzione e una per il gioco) ■ saranno anche degli effetti sonori. A meno di clamorosi risvolti espistolari, la routine che utilizzeremo sarà quella ge-





nerata dall'editor musicale descritto in «Scuola di videogame». Per la presentazione sono ancora molto indeciso e non oso pronunciarmi: se non l'avete capito, sto aspettando i vostri consigli. La storia per il Megagame credo sia già stata inviata da molti di voi, ma in ogni modo voglio farvi ascoltare la mia: «Un errore. Un banale ma tragico errore bastò ■ rendere poltiglia il frutto di 5 miliardi di anni di evoluzione. Ma erano stati in molti a supporre la probabilità di una simile tragedia ■ fortunatamente non tutto era andato perduto. Molto tempo è passato ormai e di quell'evento sembra essere rimasto solo un ricordo, registrato in qualche luogo, non in una mente umana. Quasi per volontà del destino ci si ritrova, dopo 3000 anni in quelle stesse crudeli condizioni: due grandi potenze, un obiettivo da raggiungere...». D'accordo, non sono granché come scrittore, ma era solo per farvi avere un'idea...

Scuola di videogame

L'editor musicale

seconda parte

Vediamo le caratteristiche tecniche della routine IRQ. Essa usufruisce di un max virtuale di 256 blocchi da 64 spazi ciascuno.

Ogni spazio occupa 2 byte; il primo comunica la frequenza della nota, o meglio un valore che permette di ricavarne, dopo opportuna decodifica, i giusti low ■ high byte da dare in pasto al SID, mentre il secondo ci dice lo strumento (0-15) e l'effetto (0-15) che dovremo utilizzare unitamente al valore della frequenza.

Le tracce sono 256 ed ognuna è composta da 3 blocchi che possono essere scelti a piacere tra i 256 disponibili, quindi ogni traccia occupa 3 byte. Le tracce non vengono suonate dalla prima all'ultima, ma seguono le direttive dell'«indicatrice di pezzi». Quest'ultima è, in sostanza, una ulteriore struttura dati, sviluppata al fine di aumentare la durata dei brani (notato il plurale?... tra breve vediamo cosa vuol dire); ogni singolo elemento di tale struttura ci indica una traccia di partenza e una di arrivo, più la «disposizione» del set di 16 strumenti, da utilizzare per questo gruppo di tracce. Ciò vuol dire che possiamo suonare, ad esempio, dalla traccia 0 alla 20, cambiare la «disposizione» degli strumenti, quindi suonare dalla traccia 5 alla 50, cambiare ancora e suonare dalla 0 alla 100 ecc. Ma cosa si intende per «disposizione» degli strumenti? Il trucco è molto banale; ogni strumento è associato con un numero (0-15) e in fase di edit dei blocchi ciò ci permette di selezionare quello desiderato (supponiamo di aver scelto un pianoforte, al quale è associato il numero 1).

L'«indicatrice di pezzi» ci permette di alterare tale associazione, ovvero possiamo fare in modo che al numero 1 non corrisponda più il pianoforte ma, ad esempio, un violino. La stessa traccia che precedentemente sarebbe stata suonata con il piano ora verrà suonata con il violino.

L'indicatore è composto da 100 elementi, quindi da 1000 byte, considerato che per ogni elemento ne occorrono due per le tracce, di partenza e di fine, e 8 per riassociare i 16 strumenti.

Infine si possono definire un max di 10 brani distinti, selezionabili dall'utente, da comporre utilizzando i 100 elementi visti sopra. Ad esempio, il primo brano potrebbe essere composto ■ partire dal pezzo 0 fino al pezzo 70, il secondo dal 71 al 90 ecc. I brani non devono essere necessariamente 10; se vogliamo possiamo anche realizzare un unico brano composto da tutti e cento i pezzi!

Una volta avviata l'IRQ, che si «accoda» ad eventuali modifiche già effettuate, la routine entra in stato di attesa. Quando verrà depositato, in un'opportuna locazione, il valore del brano che vogliamo eseguire essa resetterà alcuni puntatori quindi inizierà a suonare. Il volume è controllabile indipendentemente, in modo diretto. In qualunque momento dell'esecuzione è possibile cambiare brano ■ «spegnere» la routine. È inoltre possibile escludere una ■ più voci per attivare, eventualmente, effetti sonori tramite routine esterne.

E fin qui specifiche che riguardano la routine vera ■ propria, che sarà inserita nel nostro programma-game. Parliamo ora dell'editor o meglio torniamo a parlarne, visto che il discorso era già cominciato lo scorso mese. Ho ritenuto opportuno inserire una nuova importantissima opzione.

Essa permette di definire un buffer pieno di valori di note messi in sequenza, ma senza l'indicazione della pausa tra una nota e l'altra. Riempito a dovere tale buffer (magari per tentativi, se non

Schema della struttura dati della routine musicale

1000 byte: elenco pezzi
0 traccia start
1 traccia end
2 strumento 1/2
3 strumento 3/4
— — —
8 traccia start
9 traccia end
10 strumento 1/2
— — —
999 strumento 15/16

256X3 byte: elenco tracce
0 blocco 0 voce 1
1 blocco 0 voce 2
2 blocco 0 voce 3
3 blocco 1 voce 1
— — —
767 blocco 255 voce 3

256X128 byte: elenco blocchi
0 nota 0 blocco 0
1 strum./eff. 0
2 nota 1 blocco 1
3 strum./eff. 1
— — —
125 ultima nota blocco 0
127 ultimo strum./eff. blocco 0
128 nota 0 blocco 1

siete dei bravi musicisti) possiamo effettuare il trasferimento in uno dei blocchi disponibili; si fa partire l'edit in modo REC-PLAY (quindi con il blocco che scorre ■ velocità costante e con l'accompagnamento delle altre due voci) e si preme il tasto «ritmo» ogni volta che vogliamo trasferire una nota dal buffer al blocco; con questo semplice accorgimento si sono separate la sessione di inserimento note e quella di inserimento pause (per la gioia di chi non sa suonare).

Premendo il tasto shift in fase di edit è possibile passare da una tastiera musicale ad ottave basse ad un'altra ad ottave alte; in tale modo si rimedia alla limitata «estensione» della tastiera del computer.

È possibile caricare e salvare il singolo strumento, l'insieme dei 16 strumenti, le tracce e i blocchi, oppure il tutto unitamente alla routine IRQ.

Ho eliminato la gestione delle opzioni tramite joystick perché era poco pratica, quindi ho associato ogni opzione ad un tasto, da premersi unitamente al tasto Commodore.

Esaminiamo come si crea uno strumento. È possibile variare il ciclo ADSR a piacimento (il release viene attivato quando si incontra uno STOP nel blocco) e così è anche per il tipo di forma d'onda, il tipo di filtro, la risonanza e la frequenza di taglio. In più sono disponibili come extra la modulazione ■ frequenza ed ampiezza variabile della frequenza della nota, della frequenza di taglio del filtro e del duty-cicle per l'ondata rettangolare, opzioni queste che aggiungono «profondità» al suono del SID. Ovviamente tutto ciò che riguarda i filtri non è indipendente da voce ■ voce, quindi se più strumenti ne fanno uso, settandoli in modo differente, il valore assegnato ai parametri relativi sarà quello dell'ultimo strumento che li ha utilizzati. Ad ogni strumento è possibile associare un nome al fine di poterlo identificare rapidamente. Veniamo all'editor di effetti. Innanzi tutto chiariamo cosa si intende per effetti.

Questi altro non sono che delle modifiche da apportare alla normale gestione dello strumento, quindi comprendono il passaggio «fluido» da una nota ad un'altra, la mancata attivazione del ciclo di ADSR, l'incremento della frequenza di modulazione dei parametri modulabili ecc. Questi «effetti» possono essere combinati insieme al fine di ottenere 16 effetti diversi...

Diamo ora un breve sguardo al funzionamento della routine musicale. Una volta avviata e una volta selezionata una delle musiche disponibili (0=nessuna musica, da 1 a 10= musiche valide) il numero fornito viene decodificato da un'apposita tabella in pezzo iniziale e pezzo finale, quindi i rispettivi puntatori vengono posizionati, inoltre si resettano alcuni parametri. Si prende con il puntatore la prima traccia (di start) ■ la seconda (di end) e si settano i 16 strumenti (con i prossimi ■ byte). Il puntatore alle tracce ci consente di posizionare i tre puntatori ai blocchi (3 byte). Si salta quindi alla routine che viene eseguita ad ogni ciclo IRQ e cioè a quella che attende che la pausa tra uno spazio e il successivo sia terminata. La prima pausa è nulla quindi si prendono subito i primi valori tramite i puntatori ai blocchi e si ricavano la frequenza e lo strumento+effetto di ciascuna voce, sempre che il flag che ne indica l'attivazione lo consenta. Se si tratta di uno spazio vuoto si passa alla voce successiva, mentre se ■ tratta di uno STOP si attiva il Release della voce. Terminato questo aggiornamento, viene resettato il contatore della pausa tra gli spazi (che ci permette di accelerare l'esecuzione del brano senza alterare la velocità di esecuzione della routine IRQ) quindi vengono aggiornati i puntatori. Quando alla ennesima IRQ i puntatori saranno usciti dal range, ovvero quando i blocchi saranno terminati, ci sarà l'aggiornamento della traccia e quando anche le tracce saranno terminate ci sarà l'aggiornamento del pezzo e così via. Dopo l'impostazione della frequenza, del tipo di strumento e del tipo di effetto, si salta alla routine che effettua appunto la gestione di tale elementi, ed anch'essa viene eseguita per ogni ciclo IRQ.

A questo punto non mi resta che comunicarvi quando l'editor sarà pronto e come sarà possibile entrarne in possesso; in effetti al momento attuale l'editor non è ancora disponibile, ma lo sarà il mese prossimo e per riceverlo a casa vostra dovrete seguire la consueta modalità per l'acquisto dei programmi.

A presto.

Megaposta

Caro Pesciolone, seguo le tue vicende dal numero di dicembre '87 (il primo) e devo dire che tutto mi aspettavo tranne che questa storia sarebbe durata per così tanto tempo. Non voglio farti un rimprovero, anzi! Sono contento che uno sforzo come il tuo è stato apprezzato dai lettori, visto che nessuno si è mai lamentato eccessivamente della lentezza e del resto non potresti fare diversamente. Io mi ritengo un discreto programmatore e sarei intenzionato a collaborare. Tu dirai: «ma perché non ti sei fatto vivo prima». La risposta è semplice: volevo farti concorrenza e avevo organizzato un gruppo per conto mio. Ci siamo basati in parte sulle tue (vostre) idee e abbiamo realizzato alcune routine in Basic che comunque sono rimaste senza seguito. Ora ti offro il mio aiuto e la mia esperienza in campo SPAZIALE visto che mi sono reso conto che da soli non possiamo farcela...
The best Thunder, Milano

Come ho più volte sottolineato, l'aiuto dei lettori è sempre il benvenuto, anche in casi, come questo, di eccessivo ritardo.

Carissimo Marco, ti scrivo per dirti alcune cosette a riguardo della validità del Megagame 64. Pur approvando quanto è stato fatto, secondo il mio modesto parere si poteva dare di più e la colpa di ciò sta nel fatto che è mancato l'incentivo a migliorarsi. Mi spiego meglio; se il lettore che forniva il miglior consiglio o la migliore routine veniva premiato con, ad esempio, un abbonamento a MC penso che in molti si sarebbero sforzati per accaparrarsi il premio. Ormai penso che sia troppo tardi per rimediare. Ti dico questo perché da parte mia non ho avuto nessuna motivazione a mandarti il frutto del mio ingegno (modestia a parte) in cambio della semplice pubblicazione di un nominativo...
Giulio Cellitti, Civitavecchia (Roma)

Mi dispiace per te, caro Giulio, ma la filosofia che abbiamo seguito è stata una scelta ben precisa. Del resto credo che la «pubblicità», conseguente proprio alla pubblicazione di un nominativo, sia molto più remunerativa di un abbonamento o comunque di una qualunque altra ricompensa, per lo meno quando si è agli inizi (non si vive di sola gloria, giustamente).

di Marco Pesce

Scuola di videogame

L'editor musicale: la routine IRQ

Sorpresi? Niente paura il «videogioco tutto nostro» ha solo preso una puntata di ferie, almeno dal punto di vista del titolo iniziale, visto che in questo numero le due sezioni del Megagame 64 sono fuse insieme. Infatti parleremo del listato che trovate in queste pagine che, oltre ad essere la routine fondamentale dell'editor musicale delle scorse due puntate, sarà inserita nel megagame per realizzare la colonna sonora

Primo avvertimento da farvi è: la routine è perfettamente (...forse sono stato troppo ottimista) funzionante, ma manca di alcune parti, come la modulazione del DUTY-CICLE e gli effetti «speciali», tuttavia può essere utilizzata anche così come ve la presento; in seguito vi comunicherò le aggiunte da fare... sempre che ciò vi interessi. Cerchiamo di non creare confusione ■ cominciamo dall'inizio.

Innazitutto vediamo come è stato scritto il listato, ovvero quale Assembler ho usato; il suo nome, almeno a detta della scritta di partenza, è Assembler V3 di Yves Han. Si lancia con un SYS 4408 e lascia ■ disposizione dell'utente 36324 byte liberi. È un Assembler molto pratico perché fa diretto utilizzo dell'editor del Basic, ovvero possiamo utilizzare istruzioni in LM come se fossero istruzioni Basic. È possibile quindi utilizzare variabili per le etichette e altri comandi come l'OLD o l'AUTO che ci permettono, rispettivamente, di recuperare un programma cancellato con NEW e di avere la numerazione automatica delle linee del listato, ad ogni return; altre possibilità sono l'utilizzo di nomi per variabili ed etichette lunghi fino ad 8 caratteri e di cifre in esadecimale (precedute da $) o binarie (precedute da %). Se non possedete un Assembler di questo tipo la digitazione del listato si presenta molto tediosa.

Le linee dalla 100 alla 120 sono le direttive per assemblare tutto ciò che si trova tra queste e il comando NEXT PASS (linea 14000).

La routine si avvia, una volta assemblata, con SYS 28672 ($7000) ■ si «accoda» ad eventuali modifiche della IRQ, visto che, una volta terminata la sua esecuzione, salta al valore precedentemente contenuto nelle locazioni $0314 e $0315 e non direttamente a $EA31.

Dalla linea 200 in poi troviamo la routine musicale vera e propria. Nella locazione indicata dall'etichetta MUSATT (d'ora in avanti diremo semplicemente il nome dell'etichetta) bisogna depositare un valore da 1 ■ 10, ovvero scegliere la musica da attivare. Lo zero equivale a nessuna musica e in tal caso si salta alla OLDIRQ.

Depositato un numero valido (la routine non controlla se il numero inserito è maggiore di 10) nel caso esso sia differente dall'ultima IRQ (UMUSATT indica il valore che aveva la MUSATT all'IRQ precedente) si attiva la nuova musica saltando a NEWMUS. Si codifica quindi il valore fornito in pezzo iniziale ■ pezzo finale, servendosi della tabella MUSICA che è stata precedentemente ordinata dall'utente con una serie di 20 valori (o meno, se le musiche sono minori di 10), che indicano appunto il pezzo iniziale e il pezzo finale di ogni musica.

L'assegnazione dei valori alle LABEL del listato avviene ■ partire dalla linea 13000. Decodificati il PPEZZO e il CPEZZI si passa all'identificazione della traccia di start (STRACC) ■ del numero di tracce da eseguire (ETRACC) ■ si effettua la riassociazione dei 16 strumenti. Vediamo come deve essere preparata questa struttura dati.

A partire dalla locazione DATI e per le 99 successive troviamo tutte le STRACC; da DATI+100 ■ DATI+199 tutte le ETRACC; da DATI+200 a DATI+299 tutti i primi e secondi strumenti (rispettivamente nel nibble basso e nel nibble alto) ■ così via fino al 15esimo e 16esimo strumento. Questo per ciò che riguarda i pezzi. Con la STRACC si ricavano i tre blocchi delle voci che sono disposti nel seguente modo: da DATI+1000 ■ DATI+1255 tutti i blocchi delle voce 1; da DATI+1256 ■ DATI+1511 quelli della voce 2; da DATI+1512 a DATI+1767 quelli della terza voce. Impostata la CPAUSA ad 1, che, essendo un contatore per attendere la IRQ dell'aggiornamento dati, viene decrementata ad ogni IRQ, l'aggiornamento sarà immediato. La VPAUSA è il valore da assegnare a CPAUSA (selezionabile dall'utente) quando quest'ultimo si azzera. Dalla linea 750 in poi troviamo l'aggiornamento della prima voce. Esso avviene solo nel caso la voce sia attiva, ovvero se in VOCEATT1 c'è uno zero (per le voci 2 e 3 si controlla VOCEATT2 e VOCEATT3). Normalmente tutte le

voci sono attive. A questo punto bisogna ricavare la locazione di memoria dalla quale prelevare la nota e lo strum/eff, basandosi sul PBLOC1 (numero del blocco della voce 1). I blocchi sono disposti a partire dalla locazione DATI+1792; il primo byte di un blocco indica la nota (da 0 a 93, per coprire le otto ottave disponibili), mentre il secondo indica lo strum/eff e così via fino al 128esimo byte. Per ogni unità indicata dal PBLOC1 quindi occorre un incremento della locazione base di 128 byte. Ciò si ottiene con la seguente operazione: si prende il PBLOC1 e lo si shifta (LSR) a destra; se il CARRY è settato vuol dire che il PBLOC1 contiene un valore dispari quindi sicuramente ci deve essere un delta dalla locazione DATI+1792 di 128 nel byte basso del puntatore che stiamo costruendo, quindi in $FB viene depositato 128 ■ il valore contenuto nell'accumulatore in seguito allo SHIFT viene usato come byte alto, quindi depositato in $FC. Ora in $FB, $FC c'è il delta da aggiungere a $FC il DATI+1792/256 che essendo allineato con la «pagina» non comporta resto da aggiungere a $FB. Trasferiamo nel registro Y il valore contenuto in CBLOC (puntatore interno del blocco) e usiamo l'indirizzamento indicizzato per ottenere finalmente il valore della nota. Nell'eventualità che detto valore sia il 254 la routine lo riconosce come STOP, quindi si organizza in modo da farlo riconoscere come tale alla prossima routine di gestione delle voce; se invece il valore è il 255 allora si tratta di uno SPAZIO e la voce non deve subire aggiornamenti (si passa direttamente a quella successiva). Se infine il valore è compreso tra 0 e 93 (gli altri valori non devono essere utilizzati) vengono ricavati i LB e HB della frequenza della nota tramite dedicate tabelle (NOTAL, NOTAH). Ottenuta la nota si preleva il successivo byte che viene scomposto in strumento e in effetto. Un comportamento simile lo abbiamo per le altre due voci fin quando non arriviamo alla routine CKCONT, ■ quale serve al fine di aggiornare il CBLOC e eventualmente la STRACC, quindi il PPEZZO ecc. Se la musica è terminata si ricomincia da capo.

Siamo giunti così alla routine STRUMEFF che è appunto quella che si occupa degli aggiornamenti delle voci e della gestione delle modulazioni (...in un prossimo futuro anche degli effetti).

Vediamo la struttura dati degli strumenti. Dalla linea 13530 in poi troviamo 16 blocchi di dati che settano i 16 strumenti con parametri scelti a caso. Il primo byte di ogni blocco dati contiene l'AD, segue quindi l'SR, la forma d'onda, il DUTY-CICLE (2 byte), la frequenza di taglio del filtro (2 byte), la risonanza, i filtri da attivare, la frequenza di modulazione del filtro, quella di modulazione della frequenza-nota e quella di modulazione del duty-cicle (anche se, per ora, non è utilizzata), segue ancora l'ampiezza della modulazione del filtro, quella della frequenza e quella del duty-cicle. Il 16esimo byte è di riempimento. La struttura dati EFFDATI non è presente ma dovrebbe comprendere un blocco di 10 byte per ogni effetto. Per ogni voce ci sono due buffer che contengono lo strumento e l'effetto attuali, o meglio tutti i dati della struttura che li identifica. Vediamo come dal valore contenuto in STRUM1 (o comunque STRUM2 o STRUM3), che è compreso tra 0 e 15, si ricavano i dati da inserire nel buffer (BUFSTR1, 2 o 3). Sappiamo già che in STRUM1 non c'è il numero dello strumento da utilizzare come effettivo, ma che tale numero deve essere convertito servendosi della tabella realizzata dalla routine che seleziona i pezzi (quella vista nella parte relativa all'impostazione iniziale della musica). Ottenuto il numero reale dello strumento possiamo final-

```
100 FOR PASS=1 TO 2
110 ORG $7000
120 IF PASS=2 THEN ORG $7000,1
125 DATI=$7000+100*256
130 SEI:LDA #<OLDIRQ:STA OLDIRQ+1
140 LDA #>MARCO:STA OLDIRQ+2
150 STA $0315:LDA #<INT:STA OLDIRQ
160 LDA #<(MARCO/256)-INT(MARCO/256)*256
170 STA $0314
180 CLI
190 RTS
200 MARCO: ROUTINE MUSICALE
210 LDA UMUSATT
220 CMP MUSATT
230 BNE NEWMUS: E' UNA NUOVA MUSICA
240 CMP #0
250 BEQ OLDI:SE LA MUSICA NON E' NULLA SALTA AL BRANO
255 JMP ROUTMUS
260 OLDI:JMP (OLDIRQ)
270 NEWMUS:
280 LDA MUSATT
290 STA UMUSATT:DEE MUSATT:LDA MUSATT
300 ASL
310 TAX:INC MUSATT
320 LDA MUSICA,X
330 STA PPEZZO
340 INX
350 LDA MUSICA,X
360 STA CPEZZI:JSR AGGIORNO:JMP FINEAGG
370 AGGIORNO:LDX #PEZZO
380 LDA DATI,X
390 STA STRACC
400 LDA DATI+100,X
410 STA ETRACC
420 LDA DATI+200,X
430 STA STRUM
440 LDA DATI+300,X
450 STA STRUM+1
460 LDA DATI+400,X
470 STA STRUM+2
480 LDA DATI+500,X
490 STA STRUM+3
500 LDA DATI+600,X
510 STA STRUM+4
520 LDA DATI+700,X
530 STA STRUM+5
540 LDA DATI+800,X
550 STA STRUM+6
560 LDA DATI+900,X
570 STA STRUM+7
580 AGG:RTS:FINE ROUT AGGIORNAM.
590 LDA DATI+1000,X
600 STA PBLOC1
610 LDA DATI+1256,X
620 STA PBLOC2
630 LDA DATI+1512,X
640 STA PBLOC3
650 LDA #0
660 STA CBLOC:RTS:FINE ROUT AGGIORNAM.
670 FINEAGG:LDA #1
680 STA CPAUSA
690 ROUTMUS
700 LDX CPAUSA
710 DEX
720 BEQ AZZI: FINE DELLA PAUSA
725 STX CPAUSA:JMP STRUMEFF
730 AZZI:LDA VPAUSA:QUINDI AGGIORNA LE
740 STA CPAUSA
750 CLC:LDA #00:STA $FB
755 LDA VOCEATT:BEQ NN1:JMP VOCE2
760 NN1:LDA PBLOC1
770 LSR
780 BCC SALI:NIENTE INCREMENTO DI 128
790 STA $FC:LDA #128:STA $FB
810 JMP CINK1
820 SALI:STA $FC
830 ;ORA IN $FB:$FC C'E' IL DELTA
840 CINK1:LDX<(DATI+1792)/256:CLC:LDA #X
850 ADC $FC:STA $FC
860 LDY CBLOC
870 LDA (#FB),Y:IN A C'E' LA NOTA
880 TAX
```

NOTE

(continua a pag. 168)

(segue da pag. 167)

2

```
880 CMP #254:BEQ ESTOP:SETTA LO STOP
885 CMP #255:BEQ ESPAZIO:E' UNO SPAZIO
887 JMP SAL2
888 ESTOP:LDA #1:STA RELEA1:JMP VOCE2
889 ESPAZIO:LDA #0:STA ATTIVA1:JMP VOCE2
890 SAL2:LDA NOTAL,X
900 STA FRQ1
910 LDA NOTAH,X
920 STA FRQ1+1
930 INY
940 LDA (SFB),Y:ADESSO DECODIFICA              LO STRUMENTO E L'EFFETTO
950 AND #15
960 STA STRUM1
970 LDA (SFB),Y
980 LSR:LSR:LSR:LSR
990 STA EFF1:LDA #1:STA ATTIVA1
991 LDA BUFSTR+2:AND #254:STA $D404:JMP VOCE2
995 JMP VOCE2
1000 CKCONT:LDY #00:JSR CKCK:TYA:BNE RMSK:JMP STRUMEFF:AGGIORNA I PUNTATORI
1005 RMSK:JMP NEWMUS
1010 CKCK:INC CBLOC:INC CBLOC:LDA CBLOC
1020 CMP #128
1030 BEQ VA11
1040 RTS
1050 VA11
1060 INC STRACC:DEC ETRACC
1070 LDA ETRACC
1080 CMP #255
1090 BEQ APEZ
1100 JSR AGGTR
1110 RTS
1120 APEZ
1130 INC PPEZZO:DEC CPEZZI
1140 LDA CPEZZI
1150 CMP #255
1160 BEQ REMUSIC
1170 JSR AGGIORNO
1180 RTS
1190 REMUSIC:LDY $FF:RTS
1750 VOCE2:CLC:LDA #00:STA $FB
1755 LDA VOCEATT2:BEQ KK2:JMP VOCE3
1760 KK2:LDA FBLOC2
1770 LSR
1780 BCC SSAL1;NIENTE INCREMENTO DI 128
1790 STA $FC:LDA #128:STA $FB
1810 JMP CIKK2
1820 SSAL1: STA $FC
1830 ;ORA IN $FB,$FC C'E' IL DELTA
1840 CIKK2:X=(DATI+1792)/256:CLC:LDA #X
1850 ADC $FC:STA $FC
1860 LDY CBLOC
1870 LDA (SFB),Y;IN A C'E' LA NOTA
1880 TAX
1883 CMP #254:BEQ SESTOP;SETTA LO STOP
1885 CMP #255:BEQ SESPAZIO;E' UNO SPAZIO
1887 JMP SSAL2
1888 SESTOP:LDA #1:STA RELEA2:JMP VOCE3
1889 SESPAZIO:LDA #0:STA ATTIVA2:JMP VOCE3
1890 SSAL2:LDA NOTAL,X
1900 STA FRQ2
1910 LDA NOTAH,X
1920 STA FRQ2+1
1930 INY
1940 LDA (SFB),Y:ADESSO DECODIFICA              LO STRUMENTO E L'EFFETTO
1950 AND #15
1960 STA STRUM2
1970 LDA (SFB),Y
1980 LSR:LSR:LSR:LSR
1990 STA EFF2:LDA #1:STA ATTIVA2
1991 LDA BUFSTR2+2:AND #254:STA $D400+11:JMP VOCE3
2750 VOCE3:CLC:LDA #00:STA $FB
2755 LDA VOCEATT3:BEQ KK3:JMP CKCONT
2760 KK3:LDA FBLOC3
2770 LSR
2780 BCC SSSAL1;NIENTE INCREMENTO DI 128
2790 STA $FC:LDA #128:STA $FB
2810 JMP CIKK3
2820 SSSAL1: STA $FC
2830 ;ORA IN $FB,$FC C'E' IL DELTA
2840 CIKK3:X=(DATI+1792)/256:CLC:LDA #X
2850 ADC $FC:STA $FC
```

4

```
4027 SEVOCE2
4028 LDA VOCEATT2:BEQ VK2:JMP SEVOCE3
4029 VK2:JSR VH2:JMP SEVOCE3:VH2:LDA ATTIVA2
4030 BNE T2EST1
4040 LDA RELEA2
4050 BNE T2EST2
4060 RTS
4070 T2EST1:JMP A2TTVOCE
4080 T2EST2
4090 LDA BUFSTR2+2
4100 AND #254
4110 STA $D400+11:LDA #0:STA RELEA2
4120 RTS
4130 A2TTVOCE
4140 LDA STRUM2:LDY #0
4141 CLC:LSR:BCC O2PI1:LDY #1
4150 O2PI1:TAX:LDA STRUM,X
4155 DEY:BEQ T2AXIT
4160 ASL:ASL:ASL:ASL
4170 TAX:JMP L2P1
4175 T2AXIT:AND #240:TAX
4176 L2P1:LDY #0
4180 L2OOP1:LDA STRDATI,X
4190 STA BUFSTR2,Y
4200 INX:INY:CPY #15
4210 BNE L2OOP1
4220 LDA EFF2
4230 ASL:ASL:ASL:ADC EFF2:ADC EFF2
4240 TAX:LDY #0
4250 L2OOP2:LDA EFFDATI,X
4260 STA BUFEFF2,Y
4270 INX:INY:CPY #10
4280 BNE L2OOP2
4290 LDA #0:STA ATTIVA2:STA FLDIR2:STA NINC2
4300 LDA BUFSTR2
4310 STA $D400+12:      AD
4320 LDA BUFSTR2+1
4330 STA $D400+13:      SR
4340 LDA BUFSTR2+3
4350 STA $D400+9; D-C LOW
4360 LDA BUFSTR2+4
4370 STA $D400+10;D-C HIGH
4380 LDA BUFSTR2+5
4390 BNE N2SALTAF
4400 LDA BUFSTR2+6
4410 BEQ S2ALRIS
4420 N2SALTAF:STA $D400+22:STA FFIL+1
4430 LDA BUFSTR2+5:STA $D400+21:STA FFIL
4440 LDA BUFSTR2+12:STA AMPFIL
4445 LDA BUFSTR2+9:STA FRQFIL
4450 LDA BUFSTR2+7:ORA RLT
4470 ORA #2:STA $D400+23:STA RLT:JMP R2LT2
4480 S2ALRIS:LDA RLT:AND #253:STA RLT:STA $D400+23
4490 R2LT2:LDA BUFSTR2+8
4500 ORA VOLUME
4510 STA $D400+24
4520 LDA FRQ2
4530 STA $D400+7
4540 LDA FRQ2+1
4550 STA $D400+8
4570 LDA BUFSTR2+2:STA $D400+11:RTS
5027 SEVOCE3
5028 LDA VOCEATT3:BEQ VK3:JMP STREFF
5029 VK3:JSR VH3:JMP STREFF:VH3:LDA ATTIVA3
5030 BNE T3EST1
5040 LDA RELEA3
5050 BNE T3EST2
5060 RTS
5070 T3EST1:JMP A3TTVOCE
5080 T3EST2
5090 LDA BUFSTR3+2
5100 AND #254
5110 STA $D400+18:LDA #0:STA RELEA3
5120 RTS
5130 A3TTVOCE
5140 LDA STRUM3:LDY #0
5141 CLC:LSR:BCC O3PI1:LDY #1
5150 O3PI1:TAX:LDA STRUM,X
5155 DEY:BEQ T3AXIT
5160 ASL:ASL:ASL:ASL
5170 TAX:JMP L3P1
5175 T3AXIT:AND #240:TAX
```

3

```
2860 LDY CBLOC
2870 LDA (@FB),Y;IN A C'E' LA NOTA
2880 TAX
2883 CMP #254:BEQ SSESTOP;SETTA LO STOP
2885 CMP #255:BEQ SSESPAZIO;E' UNO SPAZIO
2887 JMP SSSAL2
2888 SSESTOP:LDA #1:STA RELEA3:JMP CKCONT
2889 SSESPAZIO:LDA #0:STA ATTIVA3:JMP CKCONT
2890 SSSAL2:LDA NOTAL,X
2900 STA FRQ3
2910 LDA NOTAH,X
2920 STA FRQ3+1
2930 INY
2940 LDA (@FB),Y;ADESSO DECODIFICA              LO STRUMENTO E L'EFFETTO
2950 AND #15
2960 STA STRUM3
2970 LDA (@FB),Y
2980 LSR:LSR:LSR:LSR
2990 STA EFF3:LDA #1:STA ATTIVA3
2991 LDA BUFSTR3+2:AND #254:STA @D400+18
3000 JMP CKCONT
3027 STRUMEFF
3028 LDA VOCEATT1:BEQ VK1:JMP SEVOCE2
3029 VK1:JSR VH1:JMP SEVOCE2:VH1:LDA ATTIVA1
3030 BNE TEST1
3040 LDA RELEA1
3050 BNE TEST2
3060 RTS
3070 TEST1:JMP ATTVOCE
3080 TEST2
3090 LDA BUFSTR+2
3100 AND #254
3110 STA @D400+4:LDA #0:STA RELEA1
3120 RTS
3130 ATTVOCE;TRASFERISCE UN NUOVO STR                    NEL BUFFER
3140 LDA STRUM1:LDY #0
3141 CLC:LSR:BCC OPI1:LDY #1
3150 OPI1:TAX:LDA STRUM,X
3155 DEV:BEQ TAXIT
3160 ASL:ASL:ASL:ASL;MOLTIP. PER 16
3170 TAX:JMP LP1
3175 TAXIT:AND #240:TAX
3176 LP1:LDY #0
3180 LOOP1:LDA STRDATI,X
3190 STA BUFSTR,Y
3200 INX:INY:CPY #15
3210 BNE LOOP1
3220 LDA EFF1
3230 ASL:ASL:ASL:ADC EFF1:ADC EFF1;              MOLTIPLICA PER 10
3240 TAX:LDY #0
3250 LOOP2:LDA EFFDAT1,X
3260 STA BUFEFF,Y
3270 INX:INY:CPY #10
3280 BNE LOOP2
3290 LDA #0:STA ATTIVA1:STA FLDIR:STA NINC
3300 LDA BUFSTR
3310 STA @D400+5;       AD
3320 LDA BUFSTR+1
3330 STA @D400+6;       SR
3340 LDA BUFSTR+3
3350 STA @D400+2; D-C LOW
3360 LDA BUFSTR+4
3370 STA @D400+3; D-C HIGH
3380 LDA BUFSTR+5
3390 BNE NSALTAF
3400 LDA BUFSTR+6
3410 BEQ SALRIS
3420 NSALTAF:STA @D400+22:STA FFIL+1
3430 LDA BUFSTR+5:STA @D400+21:STA FFIL
3440 LDA BUFSTR+12:STA AMPFIL
3445 LDA BUFSTR+9:STA FRQFIL
3450 LDA BUFSTR+7:ORA RLT
3470 ORA #1:STA @D400+23:STA RLT:JMP RLT2
3480 SALRIS:LDA RLT:AND #254:STA RLT:STA @D400+23
3490 RLT2:LDA BUFSTR+8
3500 ORA VOLUME
3510 STA @D400+24
3520 LDA FRQ1
3530 STA @D400
3540 LDA FRQ1+1
3550 STA @D401
3570 LDA BUFSTR+2:STA @D404:RTS ;          ■ INFINE     ^SETTA LA FORMA D'ONDA
```

5

```
5176 L300P1:LDY #0
5180 L300P1:LDA STRDATI,X
5190 STA BUFSTR3,Y
5200 INX:INY:CPY #15
5210 BNE L300P1
5220 LDA EFF3
5230 ASL:ASL:ASL:ADC EFF3:ADC EFF3
5240 TAX:LDY #0
5250 L300P2:LDA EFFDAT1,X
5260 STA BUFEFF3,Y
5270 INX:INY:CPY #10
5280 BNE L300P2
5290 LDA #0:STA ATTIVA3:STA FLDIR3:STA NINC3
5300 LDA BUFSTR3
5310 STA @D400+19;       AD
5320 LDA BUFSTR3+1
5330 STA @D400+20;       SR
5340 LDA BUFSTR3+3
5350 STA @D400+16; D-C LOW
5360 LDA BUFSTR3+4
5370 STA @D400+17;D-C HIGH
5380 LDA BUFSTR3+5
5390 BNE N3SALTAF
5400 LDA BUFSTR3+6
5410 BEQ S3ALRIS
5420 N3SALTAF:STA @D400+22:STA FFIL+1
5430 LDA BUFSTR3+5:STA @D400+21:STA FFIL
5440 LDA BUFSTR3+12:STA AMPFIL
5445 LDA BUFSTR3+9:STA FRQFIL
5450 LDA BUFSTR3+7:ORA RLT
5470 ORA #4:STA @D400+23:STA RLT:JMP R3LT2
5480 S3ALRIS:LDA RLT:AND #251:STA RLT:STA @D400+23
5490 R3LT2:LDA BUFSTR3+8
5500 ORA VOLUME
5510 STA @D400+24
5520 LDA FRQ3
5530 STA @D400+14
5540 LDA FRQ3+1
5550 STA @D400+15
5570 LDA BUFSTR3+2:STA @D400+18:RTS
6000 STREFF;MODULAZIONE DI FREQUENZA
6005 LDX #0:LDY #0
6010 STRLOOP:LDA FLDIR,X
6020 BNE RSBC
6030 LDA FRQ1+1,X
6040 CMP #255
6050 BNE INCREM
6060 CLC:LDA FRQ1,X
6070 ADC BUFSTR+13,Y
6080 BCC INCREM
6090 LDA #1
6100 STA FLDIR,X
6110 JMP STREFF2
6120 INCREM:CLC:LDA FRQ1,X
6130 ADC BUFSTR+13,Y:STA FRQ1,X
6140 BCC NONINC
6150 INC FRQ1+1,X
6160 NONINC:INC NINC,X
6170 LDA NINC,X
6180 CMP BUFSTR+10,Y
6190 BEQ SONOUG
6200 JMP STREFF2
6210 SONOUG:LDA #1
6220 STA FLDIR,X
6230 JMP STREFF2
6240 RSBC
6250 LDA FRQ1+1,X
6260 CMP #0
6270 BNE INCR2
6280 CLC:LDA FRQ1,X
6290 SBC BUFSTR+13,Y
6300 BCS INCR2
6310 LDA #0
6320 STA FLDIR,X
6330 JMP STREFF2
6340 INCR2:CLC:LDA FRQ1,X
6350 SBC BUFSTR+13,Y:STA FRQ1,X
6360 BCS NONDEC
6370 DEC FRQ1+1,X
6380 NONDEC
6390 DEC NINC,X
6400 CLC:LDA #0:SBC BUFSTR+10,Y
```

(continua a pag. 170)

6

(segue da pag. 169)

```
6420 CMP NINC,X
6430 BEQ INVERT
6440 JMP STREFF2
6450 INVERT:LDA #0
6460 STA FLDIR,X
6500 STREFF2
6510 INX:INX:TYA:CLC:ADC #25:TAY
6520 CPX #6:BEQ STREFF3
6530 JMP STRLOOP
7000 STREFF3;MODULAZIONE DEL FILTRO
7010 LDA F2DIR
7020 BNE ARSBC
7030 LDA FFIL+1
7040 CMP #7
7050 BNE AINCREM
7060 CLC:LDA FFIL
7070 ADC AMPFIL
7080 BCC AINCREM
7090 LDA #1
7100 STA F2DIR
7110 JMP STREFF4
7120 AINCREM:CLC:LDA FFIL
7130 ADC AMPFIL:STA FFIL
7140 BCC ANONINC
7150 INC FFIL+1
7160 ANONINC:INC N2INC
7170 LDA N2INC
7180 CMP FROFIL
7190 BEQ ASONOUG
7200 JMP STREFF4
7210 ASONOUG:LDA #1
7220 STA F2DIR
7230 JMP STREFF4
7240 ARSBC
7250 LDA FFIL+1
7260 CMP #0
7270 BNE AINCR2
7280 CLC:LDA FFIL
7290 SBC AMPFIL
7300 BCS AINCR2
7310 LDA #0
7320 STA F2DIR
7330 JMP STREFF4
7340 AINCR2:CLC:LDA FFIL
7350 SBC AMPFIL:STA FFIL
7360 BCS ANONDEC
7370 DEC FFIL+1
7380 ANONDEC
7390 DEC N2INC
7400 CLC:LDA #0:SBC FROFIL
7420 CMP N2INC
7430 BEQ AINVERT
7440 JMP STREFF4
7450 AINVERT:LDA #0
7460 STA F2DIR
7500 STREFF4
9070 MODULA
9080 LDA FRQ1:STA $D400:LDA FRQ1+1:STA $D401
9090 LDA FRQ2:STA $D400+7:LDA FRQ2+1:STA $D401+7
9100 LDA FRQ3:STA $D400+14:LDA FRQ3+1:STA $D401+14
9180 LDA FFIL:STA $D400+21:LDA FFIL+1:STA $D400+22
10000 JMP (OLDIRQ)
13000 NUMMUS: BYTE 0
13010 VOCEATT1:BYTE 0:VOCEATT2:BYTE 0:VOCEATT3:BYTE 0
13020 MUSATT: BYTE 0
13030 UMUSATT:BYTE 0
13040 PPEZZO: BYTE 0
13050 CPEZZI: BYTE 0
13060 STRACC: BYTE 0
13070 EYRACC: BYTE 0
13080 STRUM:  BYTE 0,0,0,0,0,0,0,0
13090 PBLOC1: BYTE 0
13100 PBLOC2: BYTE 0
13110 PBLOC3: BYTE 0
13115 CBLOC:  BYTE 0
13120 CPAUSA: BYTE 0
13125 VPAUSA: BYTE 10
13126 FLDIR:BYTE 0:NINC:BYTE 0
13127 FLDIR2:BYTE 0:NINC2:BYTE 0
13128 FLDIR3:BYTE 0:NINC3:BYTE 0
13129 F2DIR:BYTE 0:N2INC:BYTE 0
13130 MUSICA:
```

7

```
13140 BYTE 0,1,0,40,0,60,8,19,0,0
13145 BYTE 0,0,0,0,0,0,0,0,0,0
13150 FRQ1:   WORD 0
13160 FRQ2:   WORD 0
13170 FRQ3:   WORD 0
13175 FFIL:   WORD 0
13176 AMPFIL:BYTE 0:FROFIL:BYTE 0
13180 RELEA1: BYTE 0
13190 RELEA2: BYTE 0
13200 RELEA3: BYTE 0
13220 OLDIRQ: WORD 0
13230 ATTIVA1:BYTE 0
13240 ATTIVA2:BYTE 0
13250 ATTIVA3:BYTE 0
13260 STRUM1: BYTE 0
13270 STRUM2: BYTE 0
13280 STRUM3: BYTE 0
13290 EFF1:   BYTE 0
13300 EFF2:   BYTE 0
13310 EFF3:   BYTE 0
13315 RLT:    BYTE 0
13316 VOLUME: BYTE 15
13320 BUFSTR: BYTE 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
13330 BUFEFF: BYTE 0,0,0,0,0,0,0,0,0,0,0
13350 BUFSTR2:BYTE 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
13360 BUFEFF2:BYTE 0,0,0,0,0,0,0,0,0,0
13370 BUFSTR3:BYTE 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
13380 BUFEFF3:BYTE 0,0,0,0,0,0,0,0,0,0
13390 NOTAL:BYTE 22,39,57,75,95,116,138,161,186,212,240,14
13400 BYTE 45,78,113,150,190,231,20,66,116,169,224,27,90,156,226
13410 BYTE 45,125,207,39,133,232,81,193,55,180,56
13420 BYTE 196,89,247,158,78,10,208,162,129,109,103,112
13430 BYTE 137,178,237,59,157,20,160,69,3,219
13440 BYTE 207,225,18,101,219,118,58,39,65,138,5,181,157
13450 BYTE 193,36,201,102,237,115,78,130,20,10,106,59
13460 BYTE 130,72,147,107,218,231,156,4,40,20
13470 NOTAH:BYTE 1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2
13480 BYTE 3,3,3,3,3,4,4,4,4,5,5,5,6,6,6,7,7,8,8,9,9,10,10
13490 BYTE 11,12,13,13,14,15,16,17,18,19,20,21,23,24,26,27
13500 BYTE 29,31,32,34,36,39,41,43,46,49,52,55,58,62,65,69,73,78,82,87,92,98
13510 BYTE 104,110,117,124,131,139,147,156,165,175,185
13520 BYTE 196,208,221,234,248
13530 STRDATI
13540 BYTE $0A,04,65,00,94,00,00,240,00,02,1,20,255,0,255,0
13550 BYTE $2C,04,17,10,10,00,00,240,0,2,2,2,10,10,100,0
13560 BYTE $2F,04,65,04,04,30,50,240,32,10,10,10,10,10,10,0
13570 BYTE $2F,04,129,10,04,00,00,240,0,2,1,2,10,255,10,0
13580 BYTE $72,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13590 BYTE $09,04,17,00,00,00,00,240,0,2,2,2,10,10,10,0
13600 BYTE $07,04,33,00,00,80,10,240,32,10,1,10,10,250,10,0
13610 BYTE $29,04,17,00,00,00,00,240,0,2,2,2,10,10,10,0
13620 BYTE $29,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13630 BYTE $09,04,17,00,00,00,00,240,0,2,2,2,10,10,10,0
13640 BYTE $09,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13650 BYTE $09,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13660 BYTE $09,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13670 BYTE $09,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13680 BYTE $09,04,33,00,00,00,00,240,0,2,2,2,10,10,10,0
13690 BYTE $09,00,00,00,00,0,0,240,0,2,2,2,10,10,10,0
13830 EFFDATI
14000 NEXT PASS
15010 POKE DATI+T,0
15020 POKE DATI+T+100,255
15030 POKE DATI+T+200,0
15040 POKE DATI+T+300,1+16
15050 POKE DATI+T+400,2+32
15060 POKE DATI+T+500,3+48
15070 POKE DATI+T+600,4+64
15080 POKE DATI+T+700,5+80
15090 POKE DATI+T+800,6+96
15100 POKE DATI+T+900,7+112
READY.
```

Megaposta

Ragazzi, sono arrivate diverse lettere in risposta al mio invito di scrivere una storia per il megagame! Sono molto contento per questo, ma non posso cominciare a pubblicarne i contenuti, sia perché questa puntata è già abbastanza pienotta, sia perché voglio aspettare ancora un po'. Chissà che non ne venga fuori un nuovo Asimov...

mente trasferire con un loop (linee 3180-3210) di 16 passate i dati nel buffer. Segue la routine di trasferimento dei dati dell'effetto che utilizza il suo loop ■ 10 passate (linee 3250-3280). Ottenuti i nostri buffer possiamo effettuare il trasferimento decisivo nei registri del SID, dedicando particolare attenzione all'utilizzo dei filtri (linee 3380-3510). L'ultimo parametro è ovviamente la forma d'onda, che come noto fa «partire» il processore. Andiamo ad analizzare la subroutine STREFF, la quale si occupa delle modulazioni (linee da 6000 in poi). La prima modulazione è quella della frequenza; questa parte di routine viene eseguita 3 volte per le tre voci. La «variabile» FLDIR indica la direzione, ovvero se la frequenza deve essere incrementata o decrementata. La variabile NINC indica il numero di volte che la frequenza è stata INC/DECrementata. Il valore dell'AMPIEZZA della modulazione viene addizionato/sottratto fin quando il valore di NINC diventa pari alla FREQUENZA di modulazione (quando si decrementa il valore deve essere pari al negativo della frequenza); a tal punto si inverte il valore di FLDIR (da 0 a 1 o viceversa). Nell'eventualità che l'INC/DECremento comporti un'uscita dai limiti imposti il FLDIR verrà invertito prima di ottenere una... stonatura (in altre parole, se, ad esempio, un incremento porterebbe la frequenza della nota ad un valore più alto del massimo, non ci sarà incremento, quindi non ci sarà il passaggio alle basse frequenze). Anche per il filtro c'è la modulazione, che in questo caso è unica per le tre voci e inoltre ha un range più ristretto. La routine MODULA (9070 e seguenti) trasferisce i valori della frequenza e del taglio del filtro nei rispettivi registri SID. Da notare che se non si desidera avere modulazione basta impostare l'AMPIEZZA pari a zero. Il volume del sonoro deve essere impostato nella locazione VOLUME e non direttamente nel SID, perché altrimenti si andrebbero ad alterare i filtri impostati dalla routine. Le linee della 15010 in poi servono a creare una struttura dati iniziale per ciò che riguarda i pezzi da suonare, ma non è indispensabile (l'ho usata io per le prove). Non mi resta altro da aggiungere se non che nella prossima puntata ci occuperemo dell'editor (speriamo di poter pubblicare il listato) e delle parti mancanti. A presto. MC

IMPORTAZIONE E DISTRIBUZIONE DIRETTA PER L'ITALIA PERSONAL COMPUTERS CON

ESCLUSIVO 4 ANNI DI GARANZIA*

AREA SYSTEMS ITALIA s.r.l.

TRE SOLUZIONI AI VOSTRI PROBLEMI:

AREA SERVICE
• ASSISTENZA TECNICA E MANUTENZIONE
• IN TUTTA ITALIA
• AUTOMATICA, ROBOTICA E TELEMISURE

AREA SOFTWARE
• SOFTWARE GESTIONALE E SCIENTIFICO
• STANDARD PERSONALIZZATO
• CORSI DI FORMAZIONE

AREA TRADE
• IMPORTAZIONE DI HARDWARE SPECIFICO
• RICERCHE DI MERCATO

AREA SYSTEMS ITALIA s.r.l. - 10137 Torino
Corso Siracusa, 79 - Tel. (011) 3298580 - 351513 - Fax (011) 326872

COMPATIBILI AL 100% IBM*

MP Plus CPU 8088/2
Clock 10/12 MHz 640 Ram

MP 286 CPU 808286
Clock 10/16 MHz espandibile
fino a 4 Mb Ram in piastra madre

MP 386 CPU 80836
Clock 20/25 MHz 2Mb Ram on board

MP LCD PORTATILE
Video cristalli liquidi
elettroluminescente e a plasma
nelle versioni:
8088 - 286 - 386

A PARTIRE DA 599.000 LIRE anche a L. 29.000 mensili

* Regolare certificato di garanzia

* IBM è un marchio registrato dalla International Business Machine Corporation

RICHIEDETECI MATERIALE ILLUSTRATIVO. SCONTO PER RIVENDITORI QUALIFICATI E QUANTITÀ

di Marco Pesce

Scuola di Videogame

Incredibile ■ dirsi, l'editor musicale lo stiamo pubblicando, pezzo a pezzo, ma lo stiamo comunque pubblicando... Questo mese volevo presentarvi sia la mancante parte LM, sia la parte Basic, ma lo spazio non lo permetteva e purtroppo dovrete aspettare ancora un mese e per ora accontentarvi della parte LM

Tuttavia detta parte è molto semplice, si tratta di impostare alcune poke e di gestire i file, e qualche lettore più abile potrebbe anche cavarsela con la sola parte presentata in questa puntata. Infatti la routine che potete ammirare in queste pagine è in grado di effettuare tutte le funzioni di «EDITaggio» che vi avevo promesso nei mesi scorsi, quali appunto il CUT, COPY, PASTE o l'INSERT e il DELETE, più il monitoraggio dei blocchi in notazione semi-musicale, quindi lo scrolling dei blocchi, ad alta velocità e l'utilizzo del buffer. Ma, come al solito, procediamo con ordine.

Cominciamo col dire che il listato utilizza il solito Assembler di Yves Han ed è realizzato in modo da funzionare in abbinamento alla routine musicale del numero scorso. A tal proposito potete notare, nelle prime righe (cominciano dalla 20000 perché io ho continuato il listato... potete farlo anche voi), l'assegnazione di alcuni valori a variabili utilizzate nella routine IRQ; ciò è dovuto proprio all'interfacciamento tra le routine. È possibile che il vostro listato non sia perfettamente identico al mio (magari perché avete fatto qualche modifica) e quindi che detti valori non coincidano; in tal caso vi conviene, tanto per andare sul sicuro, effettuare un controllo. A proposito: vi avevo annunciato delle aggiunte alla routine IRQ che poi però non ho più fatto (...niente effetti, per il momento), tuttavia c'è stata una piccola modifica; occorre aggiungere le seguenti righe, che permettono il «congelamento» della routine musicale e l'azzeramento del SID:

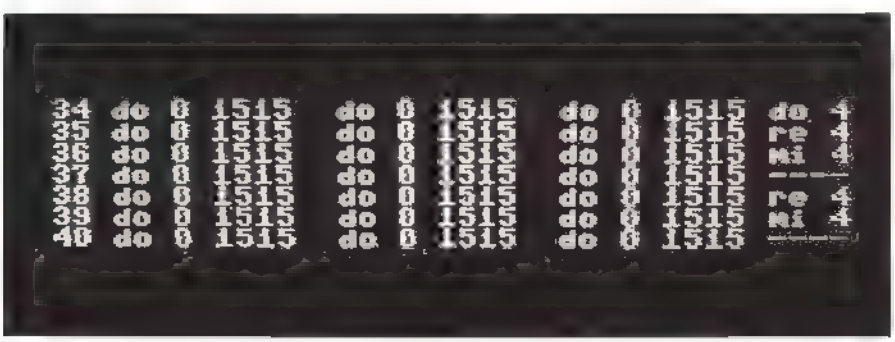
```
205  LDA CONGELA. BEQNOCONG:JMP
     (OLDIRQ)
210  NOCONG:LDA UMUSATT

13005  CONGELA:BYTE ■

260  OLDI:LDA #00:LDX #24
261  LIPPO:STA $D400,X:DEX:BNE
     LIPPO
265  JMP (OLDIRQ)
```

Detto questo torniamo al listato. La routine viene assemblata ■ partire da $6000 e si impadronisce delle IRQ e ciò vuol dire che bisogna lanciarla per prima, per poi lanciare la musicale ($7000). Un accorgimento molto importante: se fate come me, cioè se continuate la battitura del listato del mese scorso (come ho detto, non è indispensabile) ■ utilizzate l'Assembler che utilizzo io, non sarà possibile assemblare questo listato a partire da $6000 perché andrebbe ■ sovrapporsi alla variabile del Basic con conseguenze poco piacevoli; per effettuare le vostre prove conviene assemblare a partire da 50000 e poi, una volta controllato che tutto funzioni ■ dovere, cancellare la parte di listato della puntata scorsa e assemblare definitivamente a partire da $6000. A lavoro ultimato sul vostro dischetto dovrete avere la routine di questo mese e quella del mese scorso per poi farle caricare dal listato del mese prossimo; è chiaro che se volete risparmiarvi tutti questi problemi nessuno vi impedisce di richiedere il programma su dischetto...

La routine è basata quasi completamente sulla sub di visualizzazione dei blocchi. Detta sub stampa sullo schermo le 8 colonne (per sette righe) che potete ammirare in foto. La prima associa un numero di identificazione per ogni nota appartenente alla traccia che stiamo esaminando; i numeri vanno da 0 a 63 (64 note per blocco). La seconda, la quarta e la sesta contengono il valore della nota (do, re, mi ecc.); il sol è stato abbreviato per «esigenze tecniche» in SL. La terza, la quinta e la settima indicano lo strumento e l'effetto (le prime due cifre lo strumento). L'ultima colonna indica il contenuto del buffer (solo spazi e note). I tasti funzione F1 e F3 servono per lo scrolling dei blocchi mentre F5 e F7 servono per lo scrolling del buffer; scrolling che in entrambi i casi avviene alla velocità di 60 quadri al secondo (!) oppure ■ quella, utile per lasciare spazi, di un posto per ogni pressione. L'alta velocità si ottiene premendo, contemporaneamente al tasto funzione scelto, uno dei due tasti SHIFT. Per effettuare la visualizzazione la sub utilizza direttamente il contenuto dei PBLOC (1, 2, 3) ■ del CBLOC puntati dalla routine musicale; ciò vuol dire che se la routine è attiva (si disattiva ponendo un numero diverso da zero in ATTSCR) ci sarà un continuo aggiornamento dello schermo anche a musica attivata.

Il buffer è composto da 64 spazi utili ed è ciclico, ovvero quando ■ giunge all'ultimo spazio si ricomincia automaticamente dal primo. Ciò succede anche con lo scrolling, sia dei blocchi che del buffer, in entrambi i sensi (su e giù), ovvero se si giunge alla fine o all'inizio si ricomincia, rispettivamente, dall'inizio o dalla fine. La variabile MODO indica il tipo di edit che stiamo eseguendo; se posta ad 1 l'edit è di tipo REC (quindi la routine musicale deve essere disattivata), mentre se è posta ■ 2 l'edit è di tipo REC-PLAY.

La variabile ATTTAST serve per abilitare la tastiera ed inserire note in una delle due possibili destinazioni (buffer o blocco) e il flag DESTDAT decide quale deve essere la destinazione (se posto ■ 0 la destinazione è il buffer). Per inserire



le note si utilizzano i tasti seguenti (non spaventatevi):

q2w3er5t6y7u

per la prima ottava, più alcuni dei tasti successivi (in disposizione tipo tastiera musicale). Le altre ottave si ottengono inserendo nella variabile OTTAVA un delta di 12 per ogni ...ottava. Per scegliere quale strumento utilizzare si deposita nella locazione STRUMENT il valore che lo identifica. Il tasto C (che, come i tasti che seguono, si abilita solo quando in ATTTAST c'è un numero diverso da 0) serve per «tagliare» il blocco (CUT) ovvero per riempirlo di spazi (corrispondenti al valore 255). Il tasto X esegue la copia del blocco utilizzato in un buffer dedicato; questa copia potrà essere utilizzata tutte le volte che vogliamo semplicemente premendo il tasto Z, qualunque sia il blocco che stiamo utilizzando in quel momento. È chiaro che premendo nuovamente il tasto X alla copia vecchia viene sovrapposta quella del blocco attuale. L'insert e il delete si ottengono con il tasto INST/DEL e funzionano esattamente come in Basic ma operano, ovviamente, sui dati dei blocchi. Il CUT (tasto C) e il DELETE funzionano anche se stiamo operando sul buffer; funzionano solo loro perché le altre funzioni sarebbero completamente inutili. Il delete in particolare funziona in modo un po' diverso se utilizzato con il buffer, ovvero esegue la cancellazione della sola nota sulla quale si trova il puntatore, senza effettuare lo SHIFT delle altre note; potrebbe sembrare una limitazione ma non lo è, anzi. La barra spazio inserisce degli STOP che servono, come accennato più volte, ad attivare la fase di release dell'ultima nota suonata con la voce che utilizza quel blocco. Inserire STOP nel buffer è inutile ma avviene lo stesso, anche se non compare la dicitura «STOP», ma uno strano gruppo di caratteri... potete utilizzarlo per piazzare dei punti di riferimento nel buffer, visto che la colonna dei numeri di identificazione non è presente per il buffer; infatti essendo quest'ultimo a scrolling indipendente non può usufruire della stessa utilizzata dai blocchi delle tre voci.

Premendo il tasto return (tasto RITMO) si ottiene il trasferimento della nota puntata dal buffer, nello spazio puntato dal blocco. E con questo abbiamo concluso le funzioni della routine.

Le variabili utilizzate sono elencate a partire dalla riga 29000. Dalla riga 30000 in poi troviamo una piccola routine Basic che crea una tabella che verrà utilizzata dalla routine in LM. Chi proprio non sa resistere un altro mese provi a scrivere la parte in Basic da solo... ha tutte le informazioni necessarie.

La storia del Megagame

Nel numero di maggio è stato pubblicato un riquadro che invitava i lettori a scrivere una storia da inserire nell'introduzione del Megagame e già nel numero scorso si erano visti i primi risultati. Questo mese pubblichiamo quella che, secondo «l'inappellabile» giudizio di Marco Pesce è la storia da utilizzare. I complimenti vanno ad Andrea Beltrame...

«Verso la fine della 4ª età temporale si erano verificati i primi contatti con mondi che si trovavano all'Esterno della Porta Galattica. Uno dei più grandi e sviluppati era quello dei Kronos, rappresentanti di una forma di vita intelligente e quasi completamente servita da macchine sul tipo dei conosciuti e ultraperfezionati Z12.8, robot con caratteri umani, utilizzati generalmente per lavori faticosi e in avanscoperte in luoghi pericolosi. Nessuno si aspettava che i Kronos, dopo un iniziale periodo di benevola accoglienza e di scambi culturali e tecnologici, sferrassero un attacco di proporzioni inimmaginabili e si impadronissero dei principali centri di controllo intergalattico dell'Interno. Per lungo tempo si trascinò avanti una situazione di crudele sottomissione, fino a quando i popoli interni, impadronitisi dei codici di apertura del Grande Iperspazio, non riuscirono a cacciare i Kronos nelle galassie più remote. Però si presentarono subito grandi difficoltà: i crudeli dominatori avevano quasi esaurito le risorse dell'impero e si preannunciava un futuro carico di privazioni e di conflitti tra pianeti vicini. Chi avrebbe dovuto prendere il comando della situazione? Chi si sarebbe avventurato nello spazio profondo, alla ricerca di una possibilità di migliorare la vita futura?

Ora, dopo la fine dell'era dei Kronos, l'impero dell'Universo Conosciuto è scisso in due: da una parte i fedeli alla stirpe di Minar, diretto discendente dei guardiani assoluti della Porta Galattica, dall'altra i seguaci di Roman, attratti dalla sua fama di conquistatore fiero e autoritario. Chi tra i due riuscirà a ridare all'impero l'antico splendore?».

1

```
20000 PBLOC1=30357:CKCK=29014
20001 PBLOC2=30358
20002 PBLOC3=30359
20003 DAT1=$7000+10*256
20004 ATTIVA1=30406:ATTIVA2=30407:ATTIVA3=30408
20005 STRUM1=30409:STRUM2=30410:STRUM3=30411
20006 VH1=29309:VH2=29549:VH3=29789
20007 CBLOC=30360:NOTAL=30492:NOTAH=30587:FRQ1=30391:FRQ2=30393:FRQ3=30395
20090 DATS=26000
20100 FOR PASS=1 TO 3
20110 ORG $6000
20120 IF PASS=3 THEN ORG $6000,1
20130 SEI
20140 LDA #PAOLO/256
20150 STA $0315
20160 LDA #((PAOLO/256)-INT(PAOLO/256))*256
20170 STA $0314
20180 CLI
20190 RTS
20200 PAOLO:LDA ATTSCR:BEQ LLJ:JMP $EA31
20201 LLJ:JSR INSNOTE:JSR PAOLO3:JSR PAOLO4:JSR SCRBLOCC
20202 LDA #$AF:STA $FD
20203 LDA #$02:STA $FE
20204 LDA PBLOC1:STA PBLOCX:JSR PAOLO2
20205 LDA #$B6:STA $FD
20206 LDA #$05:STA $FE
20207 LDA PBLOC2:STA PBLOCX:JSR PAOLO2
20208 LDA #$C1:STA $FD
20209 LDA #$06:STA $FE
20210 LDA PBLOC3:STA PBLOCX:JSR PAOLO2:JMP $EA31
20215 PAOLO2:LDA #7
20220 STA CONTATORE
20230 SEC:LDA CBLOC
20240 SBC #6
20250 STA CBLOC7
20260 CLC:LDA #00:STA $FB
20270 LDA PBLOCX
20280 LSR
20290 BCC ZZZ1
20300 STA $FC:LDA #128:STA $FB
20310 JMP VVV1
```

(continua a pag. 192)

2 *(segue da pag. 191)*

```
20320 ZZZ1:STA $FC
20330 VVV1:X=(DATI+1792)/256:CLC:LDA #X
20340 ADC $FC:STA $FC
20350 AB3:LDA CBLOC2
20360 AND #127
20370 TAY
20380 LDA ($FB),Y
20390 STA NOTAD
20400 INY
20410 LDA ($FB),Y
20420 STA STEFF
20430 LDX NOTAD
20440 CPX #254
20450 BNE JKL1
20460 JMP STSPAZIO
20470 JKL1:CPX #255
20480 BNE JKL2
20490 JMP STSPAZIO
20500 JKL2:LDX #0
20510 LDA DATS,X
20520 STA ($FD),Y:INY
20530 LDA DATS+96,X
20540 STA ($FD),Y:INY
20550 LDA DATS+192,X
20560 STA ($FD),Y:INY
20570 LDA DATS+288,X
20580 STA ($FD),Y:INY:INY
20590 AB3:LDA STEFF
20600 AND #15
20610 TAX
20620 LDA DATS+448,X
20630 STA ($FD),Y:INY
20640 LDA DATS+384,X
20650 STA ($FD),Y:INY
20660 LDA STEFF
20670 LSR
20680 LSR
20690 LSR
20700 LSR
20710 TAX
20720 LDA DATS+448,X
20730 STA ($FD),Y:INY
20740 LDA DATS+384,X
20750 STA ($FD),Y
20760 CLC
20770 LDA $FD
20780 ADC #40
20790 STA $FD
20800 BCC AB1
20810 INC $FE
20820 AB1: DEC CONTATORE
20830 INC CBLOC2
20840 INC CBLOC2
20850 LDA CONTATORE
20860 BEQ EXIT
20870 JMP AB2
20880 EXIT:RTS
20890 STSTOP:LDY #0:LDA #19
20900 STA ($FD),Y:INY
20910 LDA #20
20920 STA ($FD),Y:INY
20930 LDA #15
20940 STA ($FD),Y:INY
20950 LDA #16
20960 STA ($FD),Y:INY:INY:JMP AB3
20990 STSPAZIO:LDY #0:LDA #45
21000 STA ($FD),Y:INY
21020 STA ($FD),Y:INY
21040 STA ($FD),Y:INY
21060 STA ($FD),Y:INY:INY:JMP AB3
21065 SFONE:LDY #0:LDA #45
21070 STA ($FD),Y:INY
21075 STA ($FD),Y:INY
21080 STA ($FD),Y:INY
21085 STA ($FD),Y:JMP ($FD)
```

4

```
21970 CMP 127:BEQ EXITD3
21980 DEX:BNE NUEXITD:RTS
21988 EXITD3:DEX:DEX:TXA:ADC OTTAVA:CMP #94:BCS NOTO
21989 TAX:STA NOTA:JMP NOTO
21990 NOTX:LDA #254:STA NOTA
21995 NOTO:LDA NOTAL,X
22000 STA FREQTAST
22010 LDA NOTAH,X
22020 STA FREQTAST+1:LDA ATTTAST:BNE CVOCUS:JMP SUONA
22030 CVOCUS:LDA VOCEUS
22035 CMP #1:BEQ BNM1
22036 CMP #2:BEQ BNM2
22037 JMP BNM3
22040 BNM1:LDA PBLOC1:STA PBLOCX:JMP MNB
22042 BNM2:LDA PBLOC2:STA PBLOCX:JMP MNB
22043 BNM3:LDA PBLOC3:STA PBLOCX
22050 MNB:LDA PBLOCX
22060 LSR
22070 BCC ZHZ1
22080 STA      DA #128:STA $FB
22090 JMP VHV1
22100 ZHZ1:STA $FC
22110 VHV1:X=(DATI+1792)/256:CLC:LDA #X
22120 ADC $FC:STA $FC
22122 LDA DESTDAT
22123 BNE SBLOCCO
22125 JMP SBUFFER
22130 SBLOCCO:LDY CBLOC
22140 LDA NOTA
22150 STA ($FB),Y
22160 INY
22170 LDA STRUMENT
22180 STA ($FB),Y
22190 LDA MODO
22200 CMP #1
22210 BNE SIAMORP
22220 JSR CECI
22230 SIAMORP:JMP SUONA
22240 SBUFFER: LDA PUBUF:AND #63:TAX
22250 LDA NOTA
22260 STA BASEBUF,X
22270 INC PUBUF
22280 SUONA
22290 LDA VOCEUS
22300 CMP #1
22310 BEQ SVOCE1
22320 CMP #2
22330 BEQ SVOCE2
22340 JMP SVOCE3
22350 SVOCE1
22360 LDA #1
22370 STA ATTIVA1
22380 LDA FREQTAST
22390 STA FRQ1
22400 LDA FREQTAST+1
22410 STA FRQ1+1
22420 LDA #0
22430 STA $D404+1
22440 LDA STRUMENT
22450 STA STRUM1
22460 JSR VH1
22470 RTS
22550 SVOCE2
22560 LDA #1
22570 STA ATTIVA2
22580 LDA FREQTAST
22590 STA FRQ2
22600 LDA FREQTAST+1
22610 STA FRQ2+1
22620 LDA #0
22630 STA $D40B+1
22640 LDA STRUMENT
22650 STA STRUM2
22660 JSR VH2
22670 RTS
      SVOCE3
```

3

```
21100 PADL03:LDA #7
21110 STA CONTATORE:LDA NO6:STA $FE:LDA #$AB:STA $FD
21120 SEC:LDA CBLOC
21130 SBC #6
21140 STA CBLOC2
21160 AC2:LDA CBLOC2
21170 AND #127
21180 LSR
21190 TAX
21200 LDY #00
21210 LDA DATS+448,X
21220 STA ($FD),Y:INY
21230 LDA DAT5+384,X
21240 STA ($FD),Y
21250 CLC
21260 LDA $FD
21270 ADC #40
21280 STA $FD
21290 BCC AC1
21300 INC $FE
21310 AC1:DEC CONTATORE
21320 INC CBLOC2
21330 INC CBLOC2
21340 LDA CONTATORE
21350 BEQ EXIT2
21360 JMP AC2
21370 EXIT2:RTS
21400 PADL04:LDA #7
21410 STA CONTATORE:LDA #06:STA $FE:LDA #$CB:STA $FD
21420 SEC:LDA PUBUF
21430 SBC #7
21440 STA PUBUF2
21460 AD2:LDA PUBUF2
21470 AND #63:TAY
21480 LDA BASEBUF,Y:CMP #255:BNE O60:JMP SPOPE
21490 O60:TAX
21500 LDY #00
21510 LDA DAT5,X
21520 STA ($FD),Y:INY
21530 LDA DAT5+96,X
21540 STA ($FD),Y:INY
21550 LDA DAT5+192,X
21560 STA ($FD),Y:INY
21570 LDA DAT5+288,X
21580 STA ($FD),Y
21600 UIY:CLC
21610 LDA $FD
21620 ADC #40
21630 STA $FD
21640 BCC AD1
21650 INC $FE
21660 AD1:DEC CONTATORE
21670 INC PUBUF2
21680 LDA CONTATORE
21690 BEQ EXIT3
21700 JMP AD2
21710 EXIT3:RTS
21800 INSNOTE
21810 LDA MODO
21820 BNE EXIT0
21830 RTS
21835 EXIT0:LDA ATTIAST:BEQ EXIT0:JSR DVOCUS:JSR EDINSDEL
21840 EXIT0:LDA 653
21850 AND #2
21860 BEQ EXIT02
21870 RTS
21880 EXIT02:LDA 197:CMP OLDTAST:BNE OSO
21885 RTS
21890 OSO:CMP #1
21900 BNE ERT
21910 STA OLDTAST:LDA PUBUF:AND #63:TAX
21930 LDA BASEBUF,X
21940 STA NOTA
21940 INC PUBUF:TAX
21960 JMP NOTA
21980 ERT:LDA OLDTAST:CMP 197:BNE WSO:RTS
21984 WSO:LDX #23:LDA 197:STA OLDTAST:CMP #60:BEQ NOTX
21985 NOEXIYO:LDA TABTAST,X
```

5

```
22760 LDA #1
22770 STA ATTIVA3
22780 LDA FRQ1AST
22790 STA FRQ3
22800 LDA FRQTAST+1
22810 STA FRQ3+1
22820 LDA #0
22830 STA $D400+18
22840 LDA STRUMENT
22850 STA STRUM3
22860 JSR VH3
22870 RTS
23030 DVOCUS:JSR DWEW:JMP VOLPE
23032 DWEW:LDA VOCEUS
23035 CMP #1:BEQ UNM1
23036 CMP #2:BEQ UNM2
23037 JMP UNM3
23040 UNM1:LDA PBLOC1:STA PBLOCX:JMP UNB
23042 UNM2:LDA PBLOC2:STA PBLOCX:JMP UNB
23043 UNM3:LDA PBLOC3:STA PBLOCX
23050 UNB:LDA PBLOCX
23060 LSR
23070 BCC DHZ1
23080 STA $FC:LDA #128:STA $FB
23090 JMP DHV1
23100 DHZ1:STA $FC
23110 DHV1:X=(DATI+1792)/256:CLC:LDA #X
23120 ADC $FC:STA $FC:RTS
23122 VOLPE:LDA DESTDAT
23123 BNE DBLOCCO
23125 JMP DBUFFER
23130 QBLOCCO:LDA 197:CMP #20:BEQ CUT
23135 CMP #23:BEQ COPY:CMP #12:BEQ PASTE:RTS
23140 CUT:LDY #127
23150 LDA #255
23160 BPBP:STA ($FB),Y
23170 DEY: BPL BPBP:RTS
23180 COPY:LDY #127
23190 BPPB:LDA ($FB),Y
23200 STA BUFFER,Y
23210 DEY
23220 BPL BPPB:RTS
23230 PASTE:LDY #127
23240 BQBQ:LDA BUFFER,Y
23250 STA ($FB),Y
23260 DEY
23270 BPL BQBQ:RTS
23280 DBUFFER
23290 LDA 197:CMP #20:BEQ C101:RTS
23300 C101:LDY #63
23310 LDA #255
23320 BQQP:STA BASEBUF,Y
23330 DEY
23340 BPL BQQP:RTS
23400 RTS:SCRBLOCC
23401 LDA 653:AND #1:BNE RE41
23402 LDA 197:CMP OLDTIST:BNE TONC:RTS
23403 TONC:STA OLDTIST
23410 RE41:LDA 197
23420 CMP #4
23430 BEQ SCRUP
23440 CMP #5
23450 BEQ SCRDW
23460 JMP SCRBUFF
23470 SCRUP
23480 LDA CBLOC
23490 BEQ PASSU
23500 DEC CBLOC
23510 DEC CBLOC
23520 JMP SCRBUFF
23530 PASSU
23540 LDA #126
23550 STA CBLOC
23560 JMP SCRBUFF
23570 SCRDW
23580 LDA CBLOC:CMP #126
23590 BEQ PASSUP
```

(continua a pag. 194)

6 *(segue da pag. 193)*

```
23600 INC CBLOC
23610 INC CBLOC
23620 JMP SCRBUFF
23630 PASSUP
23640 LDA #0
23650 STA CBLOC
23660 SCRBUFF
23661 LDA $57:AND #1:BNE RE43
23662 LDA 197:CMP OLD1ST2:BNE TONE2:RTS
23663 TONE2:STA OLDTIST2
23710 RE43:LDA 197
23720 CMP #6
23730 BEQ DSCRUP
23740 CMP #3
23750 BEQ DSCRDW
23760 RTS
23770 DSCRUP
23780 LDA PUBUF
23790 BEQ PASSU
23800 DEC PUBUF
23820 RTS
23830 PASSU
23840 LDA #63
23850 STA PUBUF
23860 RTS
23870 DSCRDW
23880 LDA PUBUF:CMP #63
23890 BEQ PASSUP
23900 INC PUBUF
23920 RTS
23930 PASSUP
23940 LDA #0
23950 STA PUBUF:RTS
24000 EDINSDEL
24010 LDA 197:CMP OLLOT:BNE OUTOT:RTS
24020 OUTOT:STA OLLOT:CMP #0
24030 BEQ CONT11
24040 RTS
24050 CONT11
24060 LDA $53
24070 AND #1
24080 BNE INST
24090 JMP DEL
24100 INST
24105 LDA DESTDAT
24106 CMP #0
24107 BNE INBF:RTS
24110 INBF:LDA CBLOC
24120 CMP #126
24130 BNE STOR
24140 RTS
24150 STOR
24160 JSR QWERT
24170 LDY #134
24180 LOOP:LDA ($FB),Y
24190 INY
24200 INY
24210 STA ($FB),Y
24220 DEY:DEY:DEY
24221 LDA ($FB),Y
24222 INY
24223 INY
24224 STA ($FB),Y
24225 DEY:DEY:DEY
24230 CPY CBLOC
24240 BEQ LOOP:INY:INY
24250 LDA #255
24260 STA ($FB),Y:INY:STA ($FB),Y
24270 RTS
24280 DEL
24290 LDA DESTDAT
24300 CMP #0
24310 BEQ DELBUF
24320 LDA CBLOC
24330 CMP #126
24340 BNE STOR2
```

7

```
24350 JSR QWERT
24360 LDA #255
24370 LDY #126
24380 STA ($FB),Y:INY:STA ($FB),Y
24390 RTS
24400 STOR2:LDY CBLOC
24410 INY:INY
24420 RANTO:LDA ($FB),Y
24430 DEY:DEY
24440 STA ($FB),Y
24450 INY:INY:INY
24451 LDA ($FB),Y
24452 DEY:DEY
24453 STA ($FB),Y
24454 INY:INY:INY
24460 CPY #128
24470 BMI RANTO
24480 LDY #126:LDA #255
24490 STA ($FB),Y:INY:STA ($FB),Y
24500 RTS
24510 DELBUF
24520 LDX PUBUF
24530 LDA #255
24540 STA BASEBUF,X
24550 RTS
29000 NOTAD:BYTE 0
29001 OLD1ST:BYTE 0
29002 OLDTIST2:BYTE 0
29003 OLLOT:BYTE 0
29005 ATTSCR:BYTE 0
29010 STEFF:BYTE 0
29020 CONTATORE:BYTE 0
29030 CBLOC2:BYTE 0
29040 PBLOCK:BYTE 0
29050 PUBUF:BYTE 0
29070 PUBUF2:BYTE 0
29080 MODO:BYTE 1
29090 NOTA:BYTE 0
29100 DESTDAT:BYTE 0
29110 FROTAST:WORD 0
29120 VOCEUS:BYTE 1
29130 STRUMENT:BYTE 1
29140 OTTAVA:BYTE 12*4
29150 TABTAST:BYTE 0,62,59,9,8,14,17,16,22,19,25,24,30,33,32,38,35,41
29160 BYTE 46,43,49,48,54
29170 ATTTAST:BYTE 0
29180 OLDTAST:BYTE 0
29190 BASEBUF=$C000
29200 BUFFER=$C000+64
29999 NEXT PASS:END
30000 DATA=26000:FOR R=0 TO 7
30005 FOR T=0 TO 11
30010 READ A:POKE T+DATA+R*12,A
30020 READ A:POKE T+96+DATA+R*12,A
30030 READ A:POKE T+192+DATA+R*12,A
30040 POKE T+288+DATA+R*12,R+48
30050 NEXT T:RESTORE:NEXT R
30200 FOR T=0 TO 63
30210 POKE T+DATA+384,L+48
30220 POKE T+DATA+384+64,H+48
30230 L=L+1:IF L>9 THEN L=0:H=H+1
30240 NEXT
40000 DATA 4,15,32
40010 DATA 4,15,35
40020 DATA 18,3,32
40030 DATA 18,3,35
40040 DATA 13,9,32
40050 DATA 6,1,32
40060 DATA 6,1,35
40070 DATA 19,12,32
40080 DATA 19,12,35
40090 DATA 12,1,32
40100 DATA 12,1,35
40110 DATA 14,4,32
```

READY.

di Marco Pesce

Scuola di videogame

Editor musicale

ultima parte

Con questa puntata abbiamo finalmente concluso i listati del nostro editor, ma prima di cominciare ad illustrarvi la funzione delle opzioni devo informarvi su alcune modifiche da apportare ai listati precedenti (per fortuna niente di complicato)

Alla riga 205 il JMP (OLDIRQ) va sostituito con JMP Streff. Alle righe 22050 e 23050 occorre la stessa modifica; inserire tra la label e l'istruzione le seguenti due istruzioni: LDA #00, STA $FB.

Possiamo cominciare...

Il listato è completamente Basic, ma chiaramente sul dischetto devono esserci anche i due file in LM delle scorse puntate; volendo potete richiedere il disco già pronto all'uso direttamente in redazione.

Le quattro foto riguardano i menu che ci aiuteranno nella creazione delle nostre colonne sonore. Dopo aver aspettato qualche secondo per il caricamento e per il reset di alcuni parametri lo schermo si presenterà come nella foto pubblicata in questa pagina. In questa sezione è possibile creare i blocchi secondo le modalità Rec+Play, Rec oppure tramite trasfermimento dal buffer. Tutte le opzioni di questa videata ■ attivano con la pressione dei tasti indicati nei pressi dell'opzione, unitamente al tasto Commodore e questa regola è valida anche per lo screen dell'edit degli strumenti; per gli screen della gestione file e dell'edit delle musiche il tasto Commodore non deve essere premuto. Questa scelta è stata fatta perché negli screen del tasto Commodore la tastiera è attiva anche in modo musicale e alcuni tasti quindi sono costretti ad avere una doppia funzione.

Esaminiamo le nostre possibilità;

— modificare la velocità di esecuzione del brano (tanto più alto è il suo valore, tanto più lenta è l'esecuzione);

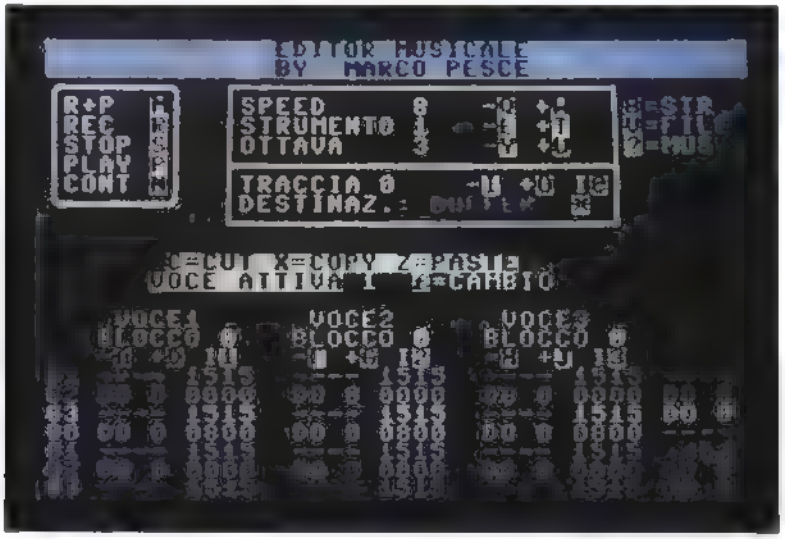
— cambiare lo strumento con il quale verranno suonate le prossime note che inseriremo in uno dei blocchi;

— cambiare l'ottava base della tastiera musicale;

— cambiare traccia;

— scegliere se depositare nei blocchi o nel buffer.

A proposito del cambio traccia, quella «i» accanto al carattere in reverse indica che con quel tasto possiamo inserire direttamente il valore della traccia desi-



È disponibile, presso la redazione, il disco con il programma pubblicato in questa rubrica completo delle parti in LM. Le istruzioni per l'acquisto e l'elenco degli altri programmi disponibili sono a pag. 263.

1

```
1 A=A+1
2 IFPEEK(52000)=170THENA=3
10 POKE 53280,0:POKE 53281,0:POKE 646,14:PRINT"(CLR)"CHR$(8);
20 GOTO 300
100 POKE198,0:WAIT198,1:POKE198,0:RETURN
150 GOSUB200:OPEN1,8,0,"$":GET#1,K$:GET#1,K$
160 GET#1,K$:GET#1,K$:IFST=64ORPEEK(197)=64THENCLOSE1:GOSUB100:RETURN
170 GET#1,LB$:GET#1,HB$:PRINTASC(LB$+CHR$(0))+256*ASC(HB$+CHR$(0));
180 GET#1,K$:IFK$<>""THENPRINTK$;:GOTO180
190 PRINT:GOTO160
200 OPEN15,8,15,"I0":CLOSE15:RETURN
300 PRINT"(RVS)            EDITOR MUSICALE           ";
310 PRINT"(RVS)            BY  MARCO FESCE           "
320 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGH
T)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)ATTENDERE PREGO..."
330 DT=31232
340 MA=30343
350 SR=30347
360 ER=30348
370 CN=30361
380 VP=30362
390 MU=30371
400 V0=30416
410 V1=29307
420 V2=29549
430 V3=29789
440 P1=30357
450 P2=30358
455 P3=30359
460 P3=30359
470 CN=30339
480 PP=30345
490 C2=30346
500 SM=30349
510 F1=30391
520 F2=30393
530 F3=30395
540 A1=30406
550 A2=30407
560 A3=30408:D=16
570 SD=30682
571 T1=30340
572 T2=30341
573 T3=30342
580 MD=25845+D
590 DS=25847+D
600 AT=25876+D
610 OT=25852+D
620 SE=25851+D
630 VS=25850+D
640 AS=25858+D
700 IFA=1THENLOAD"LM EDIT1",8,1
710 IFA=2THENLOAD"LM EDIT2",8,1
720 IFPEEK(52000)<>170THENFORT=DT+1600TODT+1791:POKET,0:NEXT:POKE52000,170
730 POKEAS,0:POKE DT,0:POKEDT+100,255
740 POKEDT+200,16:POKEDT+300,50:POKEDT+400,84
750 POKEDT+500,118:POKEDT+600,152:POKEDT+700,186
760 POKEDT+800,220:POKEDT+900,254
800 POKE AT,0:POKE MD,1:POKEVS,1:POKEDT,12*3:POKEVP,8
801 PRINT"(UP)                                                    "
805 PRINT"(WHT)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
                                  "
806 PRINT"(CYN)                                                   "
820 POKECN,1:POKEMA,1:SYS 24576
830 SYS 28672
835 PRINT"(HOME)(DOWN)(DOWN)(DOWN)"TAB(33)"(RVS)B(OFF)=STR"
836 PRINTTAB(33)"(RVS)V(OFF)=FILE"
837 PRINTTAB(33)"(RVS)?(OFF)=MUS"
840 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)VOCE1(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(
RGHT)VOCE2(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)VOCE3"
860 PRINT"(RGHT)(RGHT)(RGHT)BLOCCO(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)BLOCCO(RGHT)(RGH
T)(RGHT)(RGHT)(RGHT)BLOCCO"
```

2

```
870 PRINT"(UP)"TAB(9)PEEK(P1)
880 PRINT"(UP)"TAB(20)PEEK(P2)
890 PRINT"(UP)"TAB(31)PEEK(P3)
893 PRINT"(HOME)(WHT)(DOWN)(DOWN)  ───── ──."
894 PRINT"|R+P   (RVS)M(OFF)|"
895 PRINT"|REC   (RVS)R(OFF)|"
896 PRINT"|STOP  (RVS)S(OFF)|"
897 PRINT"|PLAY  (RVS)P(OFF)|"
898 PRINT"|CONT  (RVS)N(OFF)|"
899 PRINT"  ──────":GOSUB 1200:GOSUB1300:GOSUB1310:GOSUB1320:GOSUB1330:GOSUB1340
:GOSUB1360
900 GET A$
905 GOSUB1350:GOSUB1360:IFA$=""THEN900
906 IF (PEEK(653)AND2)=0THENPOKE198,0:GOTO 900
907 IFA$=""THEN5000
908 IFA$=""THEN6000
910 IF A$="|"THEN POKEMD,2:POKECN,0:POKEAT,1:POKEDS,1:GOSUB950:POKE1152,31
915 IF A$="┬"THEN POKEMD,1:POKECN,1:POKEAT,1:POKEDS,1:GOSUB950:POKE1192,31
920 IF A$="┐"THEN POKEMD,1:POKECN,1:POKEAT,0:GOSUB950:POKE1232,31
930 IF A$="_"THENPOKECN,0:POKEMA,0:POKEAT,0:GOSUB950:POKEMA,1:POKE1272,31
940 IF A$="|"THEN POKECN,0:GOSUB950:POKE 1312,31
945 IFC=1THENC=0:GOSUB1340:GOTO900
946 GOTO950
950 C=1:PRINT"(HOME)(WHT)(DOWN)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(
RGHT)(RGHT) (DOWN)(LEFT) (DOWN)(LEFT) (DOWN)(LEFT) (DOWN)(LEFT) ";:RETURN
960 IFA$=""ANDPEEK(VP)>1THENPOKEVP,PEEK(VP)-1:GOSUB1300:GOTO900
961 IFA$=""ANDPEEK(VP)<30THENPOKEVP,PEEK(VP)+1:GOSUB1300:GOTO900
965 IFA$=""ANDPEEK(SE)>0THENPOKESE,PEEK(SE)-1:GOSUB1310:GOTO900
966 IFA$=""ANDPEEK(SE)<15THENPOKESE,PEEK(SE)+1:GOSUB1310:GOTO900
970 IFA$=""ANDPEEK(OT)>0THENPOKEOT,PEEK(OT)-12:GOSUB1320:GOTO900
971 IFA$=""ANDPEEK(OT)<84THENPOKEOT,PEEK(OT)+12:GOSUB1320:GOTO900
975 IFA$=""ANDPEEK(DS)=1THENPOKEDS,0:GOSUB1340:GOTO 900
976 IFA$=""ANDPEEK(DS)=0THENPOKEDS,1:GOSUB1340:GOTO900
980 IFA$="▂"ANDPEEK(SR)>0THEN1010
985 IFA$="▂"ANDPEEK(SR)<255THEN1011
990 IFA$=""ANDPEEK(P1)>0THEN1050
991 IFA$=""ANDPEEK(P1)<60THEN1060
992 IFA$=""ANDPEEK(P2)>0THEN1070
993 IFA$="|"ANDPEEK(P2)<60THEN1080
994 IFA$="|"ANDPEEK(P3)>0THEN1090
995 IFA$="|"ANDPEEK(P3)<60THEN1100
996 IFA$=""THENPOKEVS,PEEK(VS)+1:IFPEEK(VS)>3THENPOKEVS,1
997 GOSUB1300
998 IFA$=""THEN2000
1000 IFA$="(ORNG)"THEN2100
1001 IFA$="(BRN)"THEN2200
1002 IFA$="(LRED)"THEN2300
1003 IFA$=""THEN5000
1009 GOTO 900
1010 POKESR,PEEK(SR)-1:POKEER,255-PEEK(SR):GOSUB 1330:GOTO1015
1011 POKESR,PEEK(SR)+1:POKEER,255-PEEK(SR):GOSUB1330
1015 POKEP1,PEEK(DT+1000+PEEK(SR))
1020 POKEP2,PEEK(DT+1256+PEEK(SR))
1025 POKEP3,PEEK(DT+1512+PEEK(SR))
1030 GOSUB1360:GOTO900
1050 POKE P1,PEEK(P1)-1:POKEDT+1000+PEEK(SR),PEEK(P1):GOSUB1360:GOTO900
1060 POKE P1,PEEK(P1)+1:POKEDT+1000+PEEK(SR),PEEK(P1):GOSUB1360:GOTO900
1070 POKE P2,PEEK(P2)-1:POKEDT+1256+PEEK(SR),PEEK(P2):GOSUB1360:GOTO900
1080 POKE P2,PEEK(P2)+1:POKEDT+1256+PEEK(SR),PEEK(P2):GOSUB1360:GOTO900
1090 POKE P3,PEEK(P3)-1:POKEDT+1512+PEEK(SR),PEEK(P3):GOSUB1360:GOTO900
1100 POKE P3,PEEK(P3)+1:POKEDT+1512+PEEK(SR),PEEK(P3):GOSUB1360:GOTO900
1200 PRINT"(HOME)(CYN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RVS)C=CUT X=COPY Z=PASTE"
1205 PRINT"(HOME)(YEL)(DOWN)(DOWN)"TAB(10)"┌────────────────────────────────┐"
1210 PRINTTAB(10)"|SPEED          -(RVS)Q(OFF)  +(RVS)W(OFF)   |"
1220 PRINTTAB(10)"|STRUMENTO      -(RVS)E(OFF)  +(RVS)T(OFF)   |"
1230 PRINTTAB(10)"|OTTAVA         -(RVS)Y(OFF)  +(RVS)U(OFF)   |"
1240 PRINTTAB(10)"├────────────────────────────────┤"
1250 PRINTTAB(10)"|TRACCIA       -(RVS)I(OFF)  +(RVS)O(OFF)  1(RVS)@(OFF)|"
1260 PRINTTAB(10)"|DESTINAZ.:            (RVS)*(OFF)  |"
1270 PRINTTAB(10)"└────────────────────────────────┘"
1280 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)    -(RVS)A(OFF)  +(RVS)D(OFF)  I(RV
S)1(OFF)"
```

(continua a pag. 172)

3 (segue da pag. 171)

```
1290 PRINTTAB(14)"(UP)-(RVS)F(OFF) +(RVS)G(OFF) 1(RVS)2(OFF)"
1295 PRINTTAB(25)"(UP)-(RVS)H(OFF) +(RVS)J(OFF) 1(RVS)3(OFF)"
1296 PRINT"(UP)(UP)(UP)(UP)(UP)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RVS)VOCE AT
TIVA(OFF)";PEEK(VS)" (RVS)E(OFF)=CAMBIO":RETURN
1300 PRINT"(HOME)(YEL)(DOWN)(DOWN)(DOWN)"TAB(20)PEEK(VF)"(LEFT) ":RETURN
1310 PRINT"(HOME)(YEL)(DOWN)(DOWN)(DOWN)(DOWN)"TAB(20)PEEK(SE)"(LEFT) ":RETURN
1320 PRINT"(HOME)(YEL)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"TAB(20)PEEK(DT)+12"(LEFT)
";RETURN
1330 PRINT"(HOME)(YEL)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"TAB(18)PEEK(SR)
" ":RETURN
1340 IFPEEK(DS)=1THENPRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN
)"TAB(22)"(WHT)BLOCCO":RETURN
1350 IFPEEK(DS)=2THENPRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)"TAB(22)"(BLU)BUFFER":RETURN
1360 PRINT"(HOME)(WHT)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"TAB(9)PEEK(P1)"(LEFT) "
1365 PRINT"(UP)(WHT)"TAB(20)PEEK(P2)"(LEFT) "
1370 PRINT"(UP)"TAB(31)PEEK(P3)"(LEFT) ":RETURN
1380 PRINT"(HOME)(YEL)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RVS)VOCE ATTIVA(OFF)";PEEK(VS
)" (RVS)E(OFF)=CAMBIO":RETURN
1400 RETURN
2000 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)INPUT TRACCIA";C$
2010 C=VAL(C$):IFC<0ORC>255THEN2000
2015 PRINT"(UP)                          "
2016 IFPEEK(197)<>64THEN2016
2017 POKEAS,0
2020 POKESR,C:POKEER,255-PEEK(SR):GOSUB1330:GOTO1015
2100 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)INPUT BLOCCO 1";C1$
2110 C1=VAL(C1$):IFC1<0ORC1>60THEN2100
2115 PRINT"(UP)                          "
2116 IFPEEK(197)<>64THEN2116
2117 POKEAS,0
2120 POKE P1,C1:POKEDT+1000+PEEK(SR),PEEK(P1):GOSUB1360:GOTO900
2200 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)INPUT BLOCCO 2";C2$
2210 C2=VAL(C2$):IFC2<0ORC2>60THEN2200
2215 PRINT"(UP)                          "
2216 IFPEEK(197)<>64THEN2216
2217 POKEAS,0
2220 POKE P2,C2:POKEDT+1256+PEEK(SR),PEEK(P2):GOSUB1360:GOTO900
2300 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)INPUT BLOCCO 3";C3$
2310 C3=VAL(C3$):IFC3<0ORC3>60THEN2300
2315 PRINT"(UP)                          "
2316 IFPEEK(197)<>64THEN2316
2317 POKEAS,0
2320 POKE P3,C3:POKEDT+1512+PEEK(SR),PEEK(P3):GOSUB1360:GOTO900
3000 PRINT"(BLK)(CLR)(LBLU)(RVS)                EDITOR MUSICALE              ";
3010 PRINT"(RVS)               BY  MARCO PESCE               (OFF)";
3012 PRINT"(CYN)(RVS)H(OFF)=MAIN MENU"
3015 POKEMD,1:POKEEN,1:POKEAT,0
3016 GOTO3300
3030 GETA$:IFA$=""THEN3030
3035 IF(PEEK(653)AND2)=0THEN3030
3040 IFA$="F"ANDKK>0THENKK=KK-16:GOTO3300
3050 IFA$="H"ANDKK<240THENKK=KK+16:GOTO3300
3060 IFA$="+"THENVF=PEEK(KK+SD):IF(VF AND240)<>0THENPOKE(KK+SD),VF-16:GOTO3300
3065 IFA$=" "THENVF=PEEK(KK+SD):IF(VF AND15)<>0THENPOKE(KK+SD),VF-1:GOTO3300
3070 IFA$="+"THENVF=PEEK(KK+SD):IF(VF AND240)<>240THENPOKE(KK+SD),VF+16:GOTO3300
3075 IFA$=" "THENVF=PEEK(KK+SD):IF(VF AND15)<>15THENPOKE(KK+SD),VF+1:GOTO3300
3080 IFA$=" "THENVF=PEEK(KK+SD+1):IF(VF AND240)<>0THENPOKE(KK+SD+1),VF-16
3085 IFA$="_"THENVF=PEEK(KK+SD+1):IF(VF AND15)<>0THENPOKE(KK+SD+1),VF-1:GOTO3400
3090 IFA$="_"THENVF=PEEK(KK+SD+1):IF(VF AND240)<>240THENPOKE(KK+SD+1),VF+16
3095 IFA$="_"THENVF=PEEK(KK+SD+1):IF(VF AND15)<>15THENPOKE(KK+SD+1),VF+1
3100 IFA$=" "THEN9000
3110 VF=PEEK(KK+SD+2):IFA$="_"ANDVF=129THENPOKEKK+SD+2,17:GOTO3300
3111 VF=PEEK(KK+SD+2):IFA$="_"ANDVF<129THENVF=(VF-1)*2+1:POKEKK+SD+2,VF:GOTO3300
3120 VF=PEEK(KK+SD+3)+PEEK(KK+SD+4)*256
3125 IFA$=" "ANDVF<3996THENVF=VF+100
3126 IFA$=" "ANDVF>99THENVF=VF-100
3127 IFA$=" "THEN3130
3128 GOTO3160
```

5

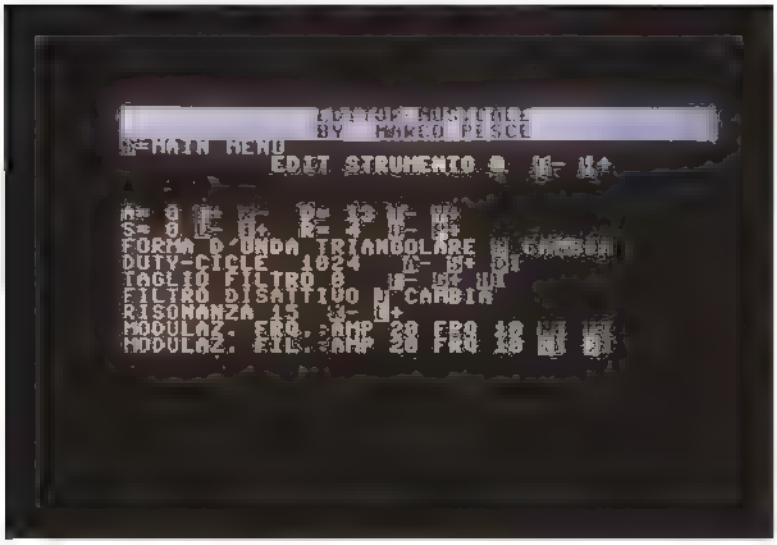
```
3471 IFPEEK(KK+SD+8)=16THENPRINT"PASS TIEF";
3472 IFPEEK(KK+SD+8)=32THENPRINT"PASS BAND";
3473 IFPEEK(KK+SD+8)=64THENPRINT"PASS HOCH";
3474 IFPEEK(KK+SD+8)=0THENPRINT"DISATTIVO";
3480 PRINT" (CYN)(RVS)3(OFF) CAMBIA(WHT)"
3490 PRINT"RISONANZA(YEL)"PEEK(KK+SD+7)/16"(CYN)(LEFT)  (RVS)K(OFF)- (RVS)L(OFF)
+(RGHT)(LEFT) "
3500 PRINT"(WHT)MODULAZ. FRQ.:(YEL)AMP"PEEK(KK+SD+9)"(LEFT) FRQ"PEEK(KK+SD+10);
3505 PRINT"(CYN)(LEFT) (RVS)N(OFF)1 (RVS)M(OFF)I(WHT) (LEFT) "
3510 PRINT"(WHT)MODULAZ. FIL.:(YEL)AMP"PEEK(KK+SD+11)"(LEFT) FRQ"PEEK(KK+SD+12);
3515 PRINT"(CYN)(LEFT) (RVS)<(OFF)I (RVS)>(OFF)I(WHT) (LEFT) "
4000 GOTO3030
5000 PRINT"(BLK)(CLR)(LBLU)(RVS)                EDITOR MUSICALE              ";
5010 PRINT"(RVS)               BY  MARCO PESCE               (OFF)";
5012 PRINT"(CYN)(RVS)V(OFF)=MAIN MENU"
5015 POKEMD,1:POKEEN,1:POKEAT,0:POKEAS,1
5020 PRINTTAB(14)"(WHT)GESTIONE FILE(DOWN)"
5025 PRINTTAB(17)"$=DIR"
5026 PRINTTAB(17)"L=LOAD"
5027 PRINTTAB(17)"S=SAVE"
5028 PRINTTAB(17)"F=FINE"
5030 GETA$:IFA$=""THEN5030
5040 IFA$="$"THENPRINT"(CLR)(YEL)DIR:":GOSUB150:GOTO5000
5050 IFA$="L"THEN5100
5060 IFA$="S"THEN5300
5065 IFA$="F"THEN9000
5066 IFA$="F"THEN5600
5070 GOTO5030
5100 PRINT"(DOWN)(DOWN)(YEL)LOAD:"
5200 GOSUB5500
5205 PRINT"(CLR)(BLK)(DOWN)(DOWN)(DOWN)LOAD"CHR$(34)B$CHR$(34)",8,1"
5206 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)POKE46,55
:RUN(HOME)"
5210 FORI=0TO9:POKE631+I,13:NEXT:POKE198,9:END
5300 PRINT"(DOWN)(DOWN)(YEL)SAVE:"
5301 GOSUB5500
5305 PRINT"(CLR)(BLK)(DOWN)(DOWN)(DOWN)POKE44,112:POKE46,168:POKE43,0:POKE45,0"
5306 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)SAVE"CHR$(34)B$CHR$(34)",8,1"
5307 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)POKE43,1:POKE44,8:POKE46,55:RUN(H
OME)"
5310 FORI=0TO9:POKE631+I,13:NEXT:POKE198,9:END
5330 GOSUB5500
5370 GOSUB5500
5500 INPUT"(WHT)NOME FILE";B$:RETURN
5600 PRINT"(YEL)FILE:"
5610 GOSUB5500:B$="S:"+B$
5620 OPEN15,8,15,B$:CLOSE15:GOTO5000
6000 POKEAS,0:PRINT"(BLK)(CLR)(LBLU)(RVS)                EDITOR MUSICALE
 ";
6010 PRINT"(RVS)               BY  MARCO PESCE               (OFF)";
6012 PRINT"(CYN)(RVS)7(OFF)=MAIN MENU"
6015 POKEMD,1:POKEEN,1:POKEAT,0:MJ=1
6020 PRINTTAB(11)"(WHT)EDIT DELLE MUSICHE"
6025 GOSUB6700
6030 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)MUSICA(YEL)"MJ"(LEFT) Q- W+
(LEFT) "
6040 PRINT"(WHT)PEZZO START(YEL)"PEEK(MU+MJ*2-2)"(LEFT) E=INS (LEFT) "
6050 PRINT"(WHT)NUMERO PEZZI(YEL)"PEEK(MU+MJ*2-1)"(LEFT) R=INS (LEFT) "
6060 PRINT"(WHT)TRACCIA(YEL)"PEEK(SR)"(LEFT) "
6070 PRINT"(WHT)PEZZO(YEL)"PEEK(PP)"(LEFT) "
6080 PRINT"(WHT)PLAY=P STOP=O"
6090 GETA$:IFA$=""THEN6030
6095 IFA$=" "THEN9000
6100 IF A$="O"THEN POKEMD,1:POKEEN,1:POKEAT,0
6110 IF A$="P"THENPOKEEN,0:POKEMA,0:POKEAT,0:POKENA,MJ
6120 IFA$="Q"ANDMJ>1THENMJ=MJ-1
6125 IFA$="W"ANDMJ<10THENMJ=MJ+1
6130 IFA$="E"THEN6700
6140 IFA$="R"THEN6800
6150 IFA$="A"ORA$="S"ORA$="D"ORA$="F"THEN6300
6200 GOTO6030
6300 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(RVS)EDIT PEZZI(OFF)"
6310 PRINT"(WHT)PEZZO(YEL)"PZ"(LEFT) (CYN)A=INS (LEFT) (YEL)"
6320 PRINT"TRACCIA START"PEEK(DT+PZ)"(LEFT) (CYN)S=INS (LEFT) (YEL)"
6330 PRINT"NUMERO TRACCE"PEEK(DT+PZ+100)"(LEFT) (CYN)D=INS (LEFT) (YEL)"
```

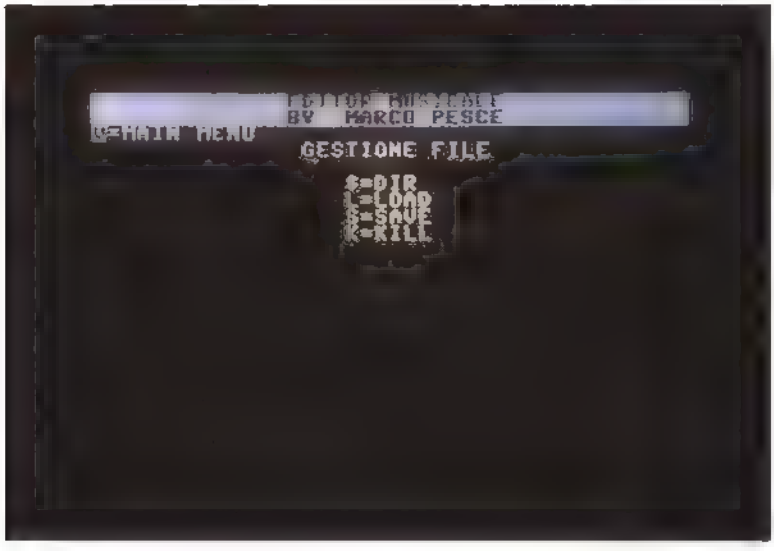
4

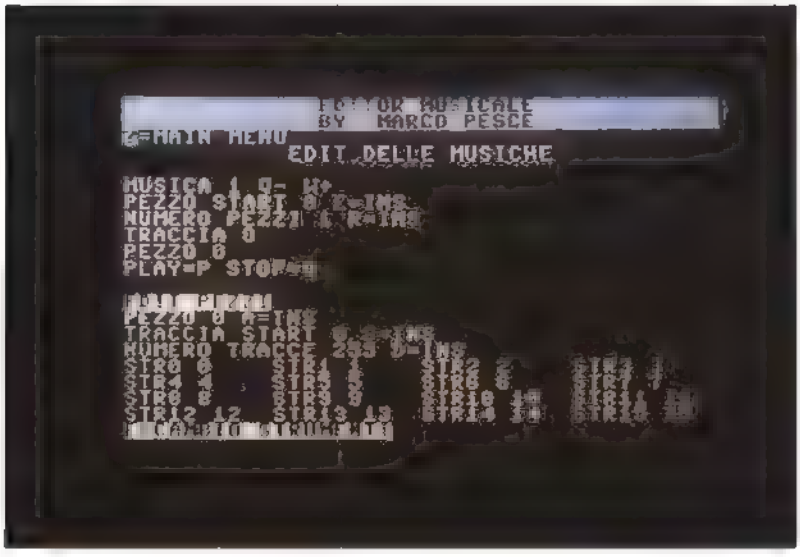
```
3130 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)DUTY-CICLE";A$:POKEAS,0
3140 VF=VAL(A$):IFVF<0ORVF>4095THEN3130
3150 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3160 POKEKK+SD+3,VF-INT(VF/256)*256:POKEKK+SD+4,VF/256
3170 VF=PEEK(KK+SD+5)+PEEK(KK+SD+6)*256
3185 IFA$="I"ANDVF<2048THENVF=VF+100
3186 IFA$="K"ANDVF>99THENVF=VF-100
3187 IFA$=""THEN3190
3188 GOTO3300
3190 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)TAGLIO FILTRO";A$:POKEAS,0
3191 VF=VAL(A$):IFVF<0ORVF>2048THEN3190
3192 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3200 POKEKK+SD+5,VF-INT(VF/256)*256:POKEKK+SD+6,VF/256
3210 VF=PEEK(KK+SD+8):IFA$="{"ANDVF=64THENPOKEKK+SD+8,0:GOTO3300
3211 VF=PEEK(KK+SD+8):IFA$="}"ANDVF=0THENPOKEKK+SD+8,16:GOTO3300
3212 VF=PEEK(KK+SD+8):IFA$="}"ANDVF<64THENVF=VF*2:POKEKK+SD+8,VF:GOTO3300
3220 IFA$="@"ANDPEEK(KK+SD+7)>0THENPOKEKK+SD+7,PEEK(KK+SD+7)-16
3225 IFA$="*"ANDPEEK(KK+SD+7)<240THENPOKEKK+SD+7,PEEK(KK+SD+7)+16
3230 IFA$=" "THEN3240
3231 IFA$=" "THEN3250
3232 IFA$=" "THEN3260
3233 IFA$=" "THEN3270
3234 GOTO3300
3240 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)FRQ:AMP";A$:POKEAS,0
3241 VF=VAL(A$):IFVF<0ORVF>255THEN3240
3242 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3243 POKE KK+SD+9,VF
3244 GOTO3300
3250 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)FRQ:FRQ";A$:POKEAS,0
3251 VF=VAL(A$):IFVF<0ORVF>255THEN3240
3252 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3253 POKE KK+SD+10,VF
3254 GOTO3300
3260 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)FIL:AMP";A$:POKEAS,0
3261 VF=VAL(A$):IFVF<0ORVF>255THEN3240
3262 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3263 POKE KK+SD+11,VF
3264 GOTO3300
3270 POKEAS,1:INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)FIL:FRQ";A$:POKEAS,0
3271 VF=VAL(A$):IFVF<0ORVF>255THEN3240
3272 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)                                                        "
3273 POKE KK+SD+12,VF
3274 GOTO3300
3300 PRINT"(HOME)(WHT)(DOWN)(DOWN)(DOWN)            EDIT STRUMENTO"KK/16"(LEFT)   (
RVS)Q(OFF)- (RVS)W(OFF)+ (LEFT) "
3325 POKESE,KK/16
3400 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)A="INT(PEEK(KK+SD)/16)"(LEF
T) (CYN)(RVS)E(OFF)+ (RVS)R(OFF)-   (WHT)D="PEEK(KK+SD)AND15"(LEFT) ";
3405 PRINT"(CYN)(RVS)T(OFF)- (RVS)Y(OFF)+ "
3410 PRINT"(WHT)S="INT(PEEK(KK+SD+1)/16)"(CYN)(LEFT) (RVS)U(OFF)- (RVS)I(OFF)+
(WHT)R=";
3411 PRINTPEEK(KK+SD+1)AND15"(LEFT) ";
3415 PRINT" (CYN)(RVS)O(OFF)- (RVS)P(OFF)+ (WHT)"
3420 PRINT"FORMA D'ONDA (YEL)";
3421 IFPEEK(KK+SD+2)=17THENPRINT"TRIANGOLARE";
3422 IFPEEK(KK+SD+2)=33THENPRINT"DENTE SEGA ";
3423 IFPEEK(KK+SD+2)=65THENPRINT"QUADRA     ";
3424 IFPEEK(KK+SD+2)=129THENPRINT"RUMORE     ";
3425 PRINT" (CYN)(RVS)@(OFF) CAMBIA(WHT)"
3430 PRINT"DUTY-CICLE (YEL)";:PRINTPEEK(KK+SD+3)+PEEK(KK+SD+4)*256"(LEFT)    ";
3440 PRINT"(CYN)(RVS)A(OFF)- (RVS)S(OFF)+ (RVS)D(OFF)I(WHT) (LEFT)    "
3450 PRINT"TAGLIO FILTRO(YEL)";:PRINTPEEK(KK+SD+5)+PEEK(KK+SD+6)*256"(LEFT)    ";
3460 PRINT"(CYN)(RVS)F(OFF)- (RVS)G(OFF)+ (RVS)H(OFF)I(WHT) (LEFT)    "
3470 PRINT"FILTRO (YEL)";
```

6

```
6340 PRINT"STR0"PEEK(DT+PZ+200)AND15"(LEFT) ","STR1"(PEEK(DT+PZ+200)AND240)/16"(
LEFT) ",
6345 PRINT"STR2"PEEK(DT+PZ+300)AND15"(LEFT) ","STR3"(PEEK(DT+PZ+300)AND240)/16"(
LEFT) ",
6350 PRINT"STR4"PEEK(DT+PZ+400)AND15"(LEFT) ","STR5"(PEEK(DT+PZ+400)AND240)/16"(
LEFT) ",
6355 PRINT"STR6"PEEK(DT+PZ+500)AND15"(LEFT) ","STR7"(PEEK(DT+PZ+500)AND240)/16"(
LEFT) ",
6360 PRINT"STR8"PEEK(DT+PZ+600)AND15"(LEFT) ","STR9"(PEEK(DT+PZ+600)AND240)/16"(
LEFT) ",
6365 PRINT"STR10"PEEK(DT+PZ+700)AND15"(LEFT) ","STR11"(PEEK(DT+PZ+700)AND240)/16
" ",
6370 PRINT"STR12"PEEK(DT+PZ+800)AND15"(LEFT) ","STR13"(PEEK(DT+PZ+800)AND240)/16
" ",
6375 PRINT"STR14"PEEK(DT+PZ+900)AND15"(LEFT) ","STR15"(PEEK(DT+PZ+900)AND240)/16
" ",
6376 PRINT"(RVS)F=CAMBIO STRUMENTI(OFF)"
6380 IFA$="S"THEN6400
6385 IFA$="D"THEN6500
6390 IFA$="A"THEN6600
6395 IFA$="F"THEN6900
6397 GOTO6030
6400 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)START";A$:A=
VAL(A$):IFA<0ORA>255THEN6400
6410 GOSUB7000:POKEDT+PZ,A
6415 A=DT+PZ
6420 IF(PEEK(A)+PEEK(A+100))>255THENPOKEA+100,255-PEEK(A)
6440 GOTO6300
6500 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)TRACCE";A$:A
=VAL(A$):IFA<0ORA>255THEN6500
6510 GOSUB7000:POKEDT+PZ+100,A
6520 IFA+PEEK(DT+PZ)>255THEN6500
6530 GOTO6300
6600 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)PEZZO";A$:A=
VAL(A$):IFA<0ORA>255THEN6600
6610 GOSUB7000:PZ=A:GOTO6300
6700 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)PEZZO START"
;A$:A=VAL(A$):IFA<0ORA>99THEN6700
6710 GOSUB7000:POKEMU+MJ*2-2,A
6715 A=MU+MJ*2-2
6720 IF(PEEK(A)+PEEK(A+1))>99THENPOKEA+1,99-PEEK(A)
6740 GOTO6030
6800 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)NUMERO PEZZI
";A$:A=VAL(A$):IFA<0ORA>99THEN6800
6810 GOSUB7000:POKEMU+MJ*2-1,A
6815 A=MU+MJ*2-2
6820 IF(PEEK(A)+PEEK(A+1))>99THENPOKEA+1,99-PEEK(A)
6840 GOTO6030
6900 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)STRUMENTO N.
";A$:A=VAL(A$):IFA<0ORA>15THEN6900
6910 GOSUB7000
6920 INPUT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)CAMBIO CON";
B$:B=VAL(B$):IFB<0ORB>15THEN6920
6930 GOSUB7000
6935 C=DT+PZ+200+100*INT(A/2)
6940 IFINT(A/2)=A/2THENPOKEC,(PEEK(C)AND240)+B:GOTO6300
6945 POKEC,(PEEK(C)AND15)+B*16:GOTO6300
7000 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DO
WN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(WHT)
                                        ":RETURN
8999 GOTO8999
9000 PRINT"(LBLU)(CLR)(RVS)            EDITOR MUSICALE         ";
9010 PRINT"(RVS)            BY  MARCO PESCE            (OFF)";
9015 POKEAS,0
9020 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)":GOTO801
10000 GOTO 10000

READY.
```







derata (tramite input) e questo vale anche per le altre opzioni che hanno le «i»; in tal modo è meno faticoso raggiungere determinati valori, considerando anche che, essendo il programma scritto in Basic, la risposta agli input non è delle più rapide (per fortuna l'input della tastiera musicale è in LM). Il gruppetto di tre opzioni visibile in alto a destra permette di passare agli altri screen (b=edit strumenti, v=gestione file, ?=edit musiche).

Passiamo allo screen per l'edit degli strumenti. Come accennato anche qui le opzioni si attivano unitamente al tasto Commodore e le modifiche ai parametri possono essere verificate in tempo reale suonando con la tastiera (cioè con la fila di tasti «QWERTY...»). Il filtro si attiva solo se il valore della frequenza di taglio è diverso da zero. Quando siamo soddisfatti del nostro nuovo set di strumenti possiamo tornare al menu principale e continuare con l'edit dei blocchi. La sezione di gestione dei file ci consente di esaminare la directory del dischetto e di cancellare (kill) file indesiderati, oltre chiaramente al salvataggio del nostro lavoro oppure al caricamente di uno lasciato in sospeso. Il file che ritroveremo sul dischetto in caso di Save è tutto quanto occorre per riascoltare le musiche a nostro piacimento anche senza l'ausilio dell'editor, ovvero comprende anche la routine IRQ che, ripetiamolo per i distratti, può essere inserita in un qualunque vostro elaborato (videogame o altro...). L'ultimo menu è quello che ci permette di «montare» il nostro film... pardon, le nostre musiche. Ognuna delle 10 musiche a disposizione deve essere composta da un «pezzo iniziale» e da un certo numero dei pezzi seguenti; ogni pezzo è composto da una traccia iniziale e da un... certo numero di quelle seguenti più da un nuovo «ordinamento» degli strumenti. Tutto qui. Ed ora eccovi un elenco delle locazioni che vi serviranno per utilizzare la routine IRQ (se per caso avete intenzione di acquistare il dischetto e quindi non le avete lette dai listati):

Volume=30416: permette di alterare il volume di riproduzione (0-15).

Musatt=30343: è la musica attiva (è possibile cambiarla in qualunque momento; 1-10).

Congela=30339: se posto ad un valore diverso da 0 sospende momentaneamente l'esecuzione del brano.

Vpausa=30362: rappresenta il numero di cicli IRQ che la routine attenderà prima di eseguire il prossimo blocco; è possibile modificarla per ottenere effetti particolari.

di Marco Pesce

Un videogioco tutto nostro

Dite la verità ne sentivate la mancanza vero? Dopo tre puntate di spudorata assenza torna ■ farsi vivo il megagame tutto nostro.
Questo mese vi illustrerò gli algoritmi del programma e le strutture dati necessarie (variabili, tabelle, flag, ecc.) in modo che tutti possano rendersi conto di quanto è stato fatto

Iniziamo questa puntata con la descrizione del programma. Il gioco dovrebbe essere ormai più che conosciuto dalla maggior parte dei lettori, ma diamone comunque in sintesi una descrizione generale.

Ci troviamo in un sistema di 16 pianeti scisso in due parti, ognuna delle quali è composta da ■ pianeti ed è governata da un capo supremo che ha il controllo totale su tutto ciò che gli appartiene. Ogni pianeta è composto, in quantità varia, di abitanti, energia, minerali, alimenti e armi; in più ci sono installazioni produttive di energia, minerali, alimenti e armi. L'estensione superficiale di ogni pianeta è variabile ■ chiaramente il suo contenuto è limitato da detto parametro. Lo scopo di ciascun capo è quello di conquistare tutti i pianeti del sistema ■ di proclamarsi quindi governatore assoluto. Ogni capo ha le seguenti possibilità:

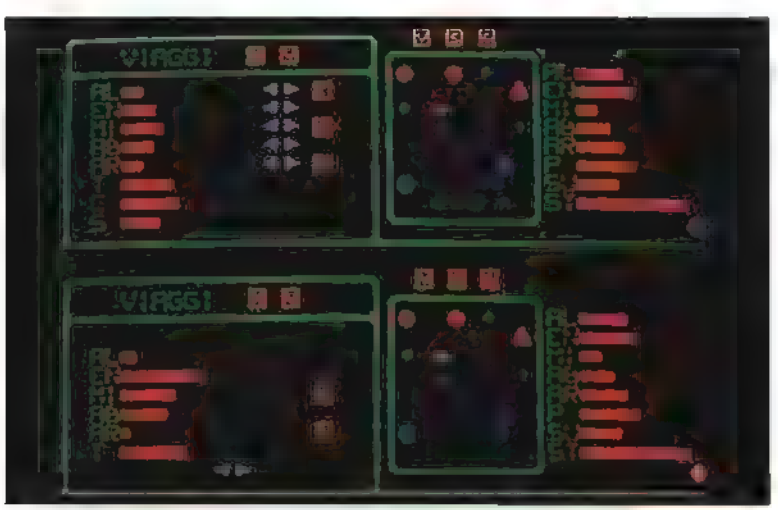
— effettuare un viaggio per lo spostamento di materiale da un pianeta ed un altro;
— costruire o distruggere installazioni produttive;
— attaccare un pianeta avversario.

Queste tre fondamentali operazioni sono coadiuvate da vari sistemi di informazione che ci permettono di conoscere lo stato dei pianeti o di eventuali viaggi o guerre in corso.

Per poter attaccare un pianeta occorrono armi ■ abitanti per pilotarle. Il giocatore dovrà quindi regolare i parametri del suo pianeta in modo da ottimizzare la produzione di questi due elementi fondamentali.

Ogni pianeta è caratterizzato da un parametro, la produttività, che stbilisce la potenza di ogni installazione produttiva. Il parametro è fisso e non varia durante il gioco (esso sostituisce i centri di ricerca e relativi parametri... piccola semplificazione).

E in sostanza il gioco è tutto qui. Si



Abbreviazioni contenute nelle schermate

AL: alimenti
EN: energia
MI: minerali
AB: abitanti
AR: armi
P: produttività o potenza armata
SU: spazio utilizzato
ST: spazio totale
P-S: potenza sorgente
P-D: potenza destinazione
T: tempo
S: spazio libero
C: costo

Tasti (arancioni)

V: viaggi o vecchio
S: strutture o sorgente
G: guerre
N: nuovo
C: conferma
D: destinazione
R: reset

Strutture dati:

Viaggi;
(Max 50+50 viaggi)
— pianeta sorgente
— pianeta destinazione
— alimenti
— armi
— minerali
— energia
— abitanti
— tempo al contatto.

Guerre;
(Max 50+50 guerre)
— pianeta sorgente
— pianeta destinazione
— quantità armi sorgente
— tempo al contatto.

Pianeta;
(16 strutture ovvero 16 pianeti)
— posizione in mappa (8 × 8)
— superficie utilizzata (oscilla tra ■ e superf. totale)
— superf. totale (oscilla tra 0 e 255)
(nota: le 9 grandezze seguenti oscillano tra 0 e 63)
— abitanti
— risorse energia
— risorse minerali
— risorse alimenti
— risorse armi
— installazioni energia
— installazioni minerali
— installazioni alimenti
— installazioni armi
— stato attuale (guerra/pace)
— possessore
— produttività (oscilla tra 0 ■ 15)
— allarme (pericolo).

Nota: gli indicatori a barra sono tutti composti da un massimo, in ampiezza orizzontale, di 63 pixel quindi le grandezze che hanno ampiezza maggiore o inferiore devono essere, rispettivamente, divise ■ moltiplicate.

Struttura del programma-base
— loop di presentazione
1 - inizializzazione totale e attivazione opzione «G» per i due giocatori
2 - test gadget mappa/pianeta trattato (2 gioc.)
3 - aggiornamento mappa (2 gioc.)
4 - test opzione attiva e JSR a detta opzione (2 gioc.)
5 - controllo viaggi
■ - controllo guerre
7 - controllo «fine del giorno» quindi aggiorn. dati
8 - controllo Game Over quindi finale
9 - JMP 2
10 - finale quindi presentazione quindi JMP 1

Nota: l'inizializzazione comprende anche lo stato iniziale dei pianeti e della suddivisione della mappa, che può essere casuale o predeterminato, cioé fisso.

Struttura gestione mappa
«Test gadget mappa/pianeta trattato»:
1 - test JOY
2 - se fire premuto test coincidenza con opzioni «V», «G» o «S», quindi attiva opzione (setta FLAG)
3 - se fire premuto test coincidenza con pianeta; quindi setta il pianeta (metti il suo numero d'ordine nel FLAG apposito)
4 - RTS
«Aggiornamento mappa»
1 - stampa cornice
2 - stampa gadget
3 - stampa pianeti con «illuminazione» di quello trattato (BLU in CIANO e ROSSO in ROSA) e con attenzione al loro stato (allarme, guerra...)
4 - stampa caratteristiche del pianeta trattato
5 - RTS

Nota imporante: quando si verifica il «cambio del giorno» devono essere resettati tutti i buffer delle opzioni seguenti perché le caratteristiche del pianeta ■ dei pianeti ai quali riferivano potrebbero essere cambiate.

Struttura opzione Pannello Viaggi
(Attenzione: il COSTO dei viaggi è funzione solo della quantità del materiale ■ corrisponde a un decremento di energia, mentre il TEMPO è funzione della sola distanza. Detti parametri possono essere definiti dal programmatore).
1 - stampa riquadro (cancella tutto e poi stampa)
2 - stampa «N» e «V» con lampeggio opportuno (o anche senza)
3 - test JOY
4 - test dei tasti «N» e «V», quindi se «N» attivalo e azzera buffer quindi setta SORG ■ DEST scelti ■ caso; se «V» attivalo
5 - test attivo «N» ■ «V» o -RTS- (se non -RTS- stampa parametri dal buffer), quindi salta a «N» o «V»
6 - «N»; test pianeti disponibili; se 1 stampa «IMPOSSIBILE» quindi «RTS»
7 - stampa «C», «S» e «D» con lampeggio opportuno
8 - test dei tasti «C», «S» e «D»; se «C» controlla che ■ sia un minimo di materiale nel buffer, quindi sottrai il costo all'energia del pianeta SORG e lo spazio, quindi imposta i parametri nella prima tabella libera e setta -RTS- al posto di «N» (quindi disattiva il lampeggio...); se «S» attiva SORG (flag); se «D» attiva DEST (flag)
9 - test «S» o «D» attivo quindi JMP «S» o «D» (per default attivo «S»)
10 - «S»; copia pianeta sorgente in pianeta trattato (per la mappa)
11 - test JOY
12 - se fire premuto; test coincidenza con pianeta e test di possibilità (ovvero, se il pianeta è dei nostri) e, solo se possibile, settalo come SORG e azzera il BUFFER
13 - JMP 17 (test frecce)

14 - «D»; copia pianeta destinazione in pianeta trattato (per la mappa)
15 - testo JOY
16 - se fire premuto; test coincidenza con pianeta ■ test di poss. quindi setta come SORG e azz. BUFFER
17 - stampa delle frecce-tasti e test con INC/DEC del buffer limitato alla disponibilità del pianeta SORG e DEC/INC corrisponde sulla strutt. del pianeta SORG e dello spazio, quindi calcolo del costo
18 - calcolo del tempo (distanza tra sorg. e dest.), quindi RTS
19 - «V»; copia dei parametri del viaggio trattato nel BUFFER
20 - stampa frecce-tasti per il cambio del viaggio
21 - test JOY
22 - test pressione delle frecce tasti, quindi cambio del viaggio trattato con spostamento del puntatore relativo sulla prossima o precedente tabella occupata
23 - gestione lampeggio alternato di «S» ■ «D» con trasferimento alternato in pianeta trattato (dalla mappa) di SORG e DEST del viaggio.

Struttura opzione Pannello Guerre
1 - stampa riquadro
2 - stampa «N» e «V» con lampeggio (o senza)
3 - test del JOY
4 - test di attivazione «N» o «V», quindi se «N» attivalo e azzera BUFFER e setta pianeta SORG ■ DEST (solo se non già presenti dall'opzione «V»); se «V» attivalo
5 - JMP «N» ■ «V»
6 - «N»; stampa parametri dal BUFFER
7 - stampa con opportuno lampeggio «C», «S» e «D»
■ - test «C», «S» o «D»; se «C» controlla che ci sia un minimo di armi ■ imposta i parametri nella prima tabella libera, quindi sottrai il costo ■ pianeta SORG e lo spazio e azzera «N» con -RTS-, quindi setta guerra al pianeta destinazione; se «S» attiva SORG; se «D» attiva DEST
9 - test attivo «S» ■ «D», quindi salta a «S» o «D»
10 - «S»; come per VIAGGI
11 - «D»; come per VIAGGI tranne che DEST deve essere dell'avversario
12 - stampa frecce-tasti
13 - test frecce-tasti e INC/DEC controllato delle armi e del costo
14 - calcolo del tempo (distanza tra i due pianeti)
15 - RTS
16 - «V»; copia nel buffer la guerra trattata
17 - test JOY
18 - test frecce-tasti per cambio guerra trattata
19 - lampeggio alternato di «S» e «D» (come per VIAGGI)
20 - RTS

Struttura opzione Pannello Strutture
1 - stampa riquadro
2 - test JOY
3 - se il pianeta trattato è diverso da quello dell'ultima «passata» allora copia pianeta trattato in buffer e azzera il costo
4 - stampa parametri dal buffer
■ - stampa «C» e «R»
■ - test JOY
7 - test attivazione «C» o «R»; se «C» premuto sottrai il costo ■ energia e sottrai spazio, quindi trasferisci il buffer in struttura dati del pianeta ■ reset del buffer con disattivazione di «N»; se «R» premuto copia il pianeta in buffer e azzera il costo (reset)
8 - gestione dei vari tasti-freccia con la seguente politica; se «decremento» e buffer > o = pianeta allora DEC costo e DEC struttura; se «decremento» e buffer < pianeta allora DEC struttura; se «incremento» e buffer > pianeta allora INC costo e INC struttura; se «incremento» ■ buffer < 0 = pianeta allora INC struttura; attenzione: se il costo (MI/EN) ■ lo spazio non sono sufficienti si blocca l'«incremento»
9 - RTS

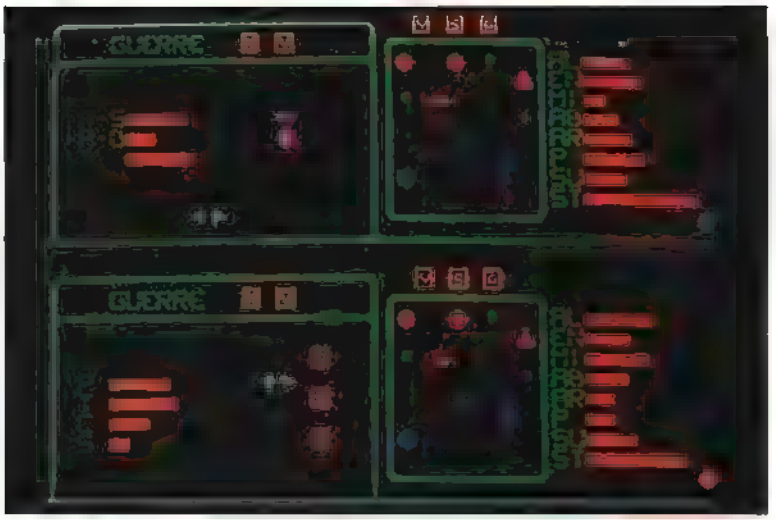
Struttura Calcolo Risorse dei Pianeti (ogni volta che termina il «giorno»)
1 - reset fine del giorno quindi salto a 2 — altrimenti — RTS-
2 - reset puntatore strutture dati dei pianeti
3 - effettua i calcoli sulla struttura trattata, quindi:
— moltiplica installazione-energia X produttività e somma il risultato a risorse-energia; se risorse-energia > 63 setta a 63; se lo spazio utilizzato è troppo occupa il massimo spazio possibile ■ decrementa le risorse-energia in eccesso, quindi non effettuare i calcoli seguenti e setta lo stato di allarme, quindi salta a 4 — (lo stesso «discorso» deve essere fatto per ogni risorsa)
— sottrai installazioni-minerali ■ risorse-energia e se < 0 azzera e setta allarme, altrimenti moltiplica inst. -min. X produtt. e aggiungi a risorse-minerali (vedi «ragionamento» del passo precedente)
— sottrai install.-aliment. a ris.-en. ■ se < 0 allarme, altrimenti sottrai install.-aliment. ■ ris.-min. e se < 0 allarme, altrimenti moltiplica e aggiungi (come al solito...)
— stesso discorso di consumi anche per le armi...
4 - calcola l'incremento (o decremento) di abitanti, ovvero raddoppia il loro numero e confrontalo con risorse-alimenti; se abitanti > ris. alim. poni abitanti=ris. alim. e effettua il giusto decremento di ris.- alim.
5 - continua con il prossimo pianeta e al termine -RTS-

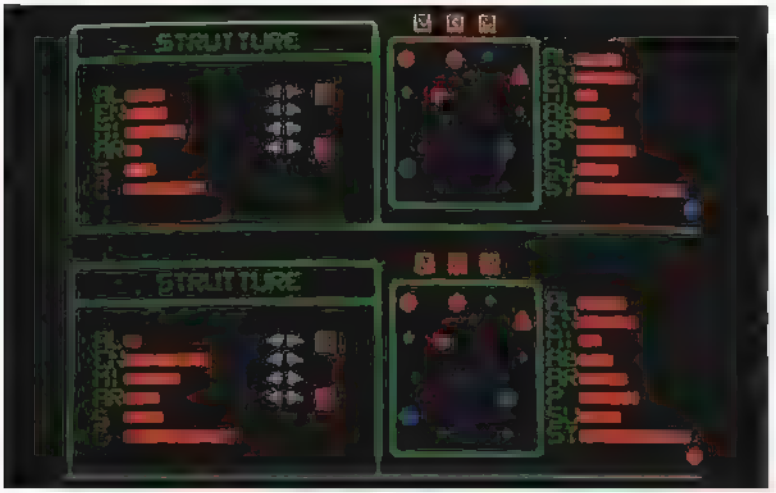
Controllo Game Over
1 - verifica che i pianeti non siano tutti dello stesso giocatore, altrimenti setta il vincitore e salta al finale
2 - RTS

Gestione dei Viaggi in corso
1 - azzera il puntatore alla tabella dei viaggi
2 - decrementa un sub-contatore e quando giunge a zero decrementa il tempo del viaggio; se giunto anch'esso a zero il viaggio è finito quindi:
— calcola lo spazio occupato dal primo materiale (e poi prosegui con i successivi), quindi rendilo (se necessario) compatibile con lo spazio, quindi verifica che lo spazio per il materiale sia sufficiente, altrimenti rendilo ancora compatibile, quindi, finalmente, aggiungi il materiale allo spazio utilizzato ■ alla risorsa interessata
— ripeti il calcolo fino ad esaurimento materiali
3 - calcola la prossima tabella; se finite -RTS-

Gestione Guerre in corso
«Guerre ancora in viaggio»:
1 - azzera il puntatore alla tabella delle guerre
2 - decrementa un sub-contatore e quando giunge a zero decrementa il tempo del viaggio delle forze (solo se > 0), per tutte le tabelle attivate.
«Guerre attive»
1 - azzera il puntatore
2 - verifica se il tempo è zero (guerra attiva) quindi ricerca eventuali strutture dati coincidenti (stesso SORG ■ DEST) ■ con tempo azzerato e somma le forze del SORG, quindi effettua lo scontro:
— calcola le armi del pianeta DEST (prendile pari a ARMI ■ ABITANTI a seconda della risorsa minore) e deposita in un buffer
— verifica che armi DEST diverse da zero, altrimenti vittoria SORG
— verifica che armi SORG diverse da zero, altrimenti vittoria

Dest
— in caso di vittoria di uno dei due contendenti, cancella guerra dalla tabella e cambia o conferma il proprietario (quindi se vittoria del SORG trasferisci le armi restanti nelle risorse del pianeta, sembra che ci sia spazio)
— effettua sub-scontro, ovvero genera un numero casuale; se dispari vittoria SORG e viceversa quindi DECrementa armi del DEST o SORG
3 - continua con il resto delle tabelle non ancora utilizzate (quelle utilizzate devono essere contraddistinte da un flag in una sub-tabella).





gioca obbligatoriamente in due (split-screen) e si manovra un cursore a testa per agire sui menu (vedi foto). Il pannello sulla destra è sempre visibile e comprende una mappa ■ una descrizione del pianeta trattato da selezionare tra quelli visibili, più tre tasti (arancioni) per selezionare una delle tre operazioni fondamentali (pannello sulla sinistra). I pianeti sono differenziati dal colore (blu-rosso, miei-tuoi). Sulla mappa sono indicati anche i pianeti in guerra (asterisco) ■ pianeti in pericolo (triangolo).

La prima foto mostra il pannello dei viaggi; premendo il tasto «N» si passa alla inizializzazione di un nuovo viaggio, quindi ■ stabiliscono le quantità da trasportare, il pianeta sorgente (tasto «S») e il pianeta destinazione (tasto «D»). Ovviamente tali pianeti dovranno essere dei nostri e nel caso tentassimo di effettuare un nuovo viaggio avendo ■ disposizione un solo pianeta, il sistema risponderà con «impossibile». Ci sarà anche un resoconto del tempo ■ del costo del viaggio, in energia (direttamente sottratta ■ quella del pianeta sorgente). Quando un'opzione è attiva il tasto corrispondente lampeggia (ad esempio «N» ed «S»). Il tasto «C» corrisponde alla conferma del viaggio; in tal caso tutti i parametri da noi stabiliti saranno salvati in una tabella ■ il computer penserà a far giungere il materiale nel tempo stabilito, da notare che quando si effettuano spostamenti come questo, viene liberato spazio sul pianeta sorgente e ovviamente sarà occupato spazio sul pianeta destinazione; a tal proposito potrebbe verificarsi che lo spazio sul pianeta destinazione non sia più sufficiente al termine del viaggio, quindi il materiale in eccesso verrà disperso nello spazio!

Visualizzare uno dei viaggi in corso (tasto «V») ci permette di conoscere il pianeta sorgente e quello destinazione, ma soprattutto il materiale trasportato; il lato economico di questa opzione è che se nel frattempo il pianeta non è più nostro avremo avvantaggiato l'avversario!

Comunque i viaggi visibili sono solo quelli che interessano i nostri pianeti o quelli che erano nostri.

Passiamo alle guerre. In caso di guerra «nuova» basta selezionare la potenza armata da inviare dal nostro pianeta sorgente (si tratta di un viaggio quindi costerà anche in termini di energia e tempo) ■ scegliere un pianeta destinazione tra quelli dell'avversario.

Le guerre «vecchie» comprendono sia quelle già attivate sia quelle in... viaggio ■ visualizzano (oltre ai soliti sorg. ■ dest.) le potenze dei due contendenti. Da notare che non esiste più possibilità di ritirata, da ambo le parti (sarebbe uno spreco di tempo e denaro visto che le nostre forze dovrebbero tornare «a casa»).

Per il pannello delle strutture ovviamente non ci sono i tasti «N» ■ «V»; basta modificare con le solite freccette e confermare con «C» (possiamo anche distruggere, per limitare i consumi ■ per riaquistare spazio), oppure effettuare un reset (tasto «R») delle modifiche fatte finora.

Megaposta

Alcuni lettori hanno inviato storie per il megagame che non c'entrano praticamente nulla con ■ struttura del gioco, ma il loro sforzo va comunque lodato (non pubblico i nominativi per evitare offese). Vi ricordo che il «servizio telefonico», che ■ era stabilito tra i lettori e Marco Pesce, non è più ■ funzione per i motivi di inutilità del suddetto. Mi scuso con tutti quelli che hanno tentato di telefonare e non mi hanno trovato.

Voglio inoltre dare una breve comunicazione a Massimiliano Pigozzi ■ Pierpaolo Bergamo di Rovigo: ho ricevuto «la merce», ma non posso dirvi altro; ne riparliamo il prossimo mese...

di Marco Pesce

Un videogioco tutto nostro

In questa puntata voglio presentarvi alcune routine in LM tratte dal Megagame, con relativa spiegazione del loro funzionamento, tutte relative alla presentazione del game

Cominciamo col dire che per la presentazione ho pensato ad uno screen con la scritta MEGAGAME 64 su uno sfondo di stelle che casualmente scintillano; al di sotto un terreno montagnoso in scrolling parallattico (vedi foto 1). Detto screen si alterna con un'altra che si differenzia solo per le scritte, che questa volta ci mostrano i nomi dei collaboratori (foto 2). In termine realizzativi occorrono routine per stampare le screen, per la commutazione delle scritte, per lo scrolling (a due strati) ■ per lo scintillio delle stelle (una alla volta). Cominciamo dallo scintillio delle stelle. Questo effetto si può realizzare facendo comparire in sequenza la serie di fotogrammi che potete osservare in foto 3, uno dopo l'altro (ovviamente) ■ in punti sparsi sull'area disponibili. Innanzi tutto diciamo che i fotogrammi sono composti ognuno da un gruppo di 6 caratteri (3x2) che ■ loro volta sono inseriti in uno dei tre set ■ disposizione del programma, per la precisione nel primo, che tra l'altro comprende anche i caratteri delle montagne e delle scritte MEGAGAME 64 ■ BY MARCO PESCE, più un set di caratteri alfabetici per le altre scritte; gli altri set sono uno per i pannelli e l'altro per lo screen di «meditazione» (vedi foto 4). Tornando allo scintillio ci accorgiamo che sono necessarie: una tabella che contiene i caratteri di ciascun fotogramma e una tabella che contiene le posizioni di stampa e i tempi di attesa tra una stampa e la successiva (ogni stampa si intende composta dall'intera sequenza di scintillio).

La prima tabella ha la seguente struttura: ogni fotogramma, che è composto da sei caratteri, è memorizzato come una serie, appunto, di sei caratteri. La seconda tabella comprende un alternarsi di byte riferiti alla pausa tra le stampe ■ alla posizione (locazione video, quindi 2 byte) di stampa. Andiamoci ad esaminare la routine delle stelle. La parte che effettua la pausa tra due stampe è integrata con ■ routine in IRQ di gestione del resto della presentazione, quindi ci occupiamo subito della parte che stampa i fotogrammi. Per prima cosa si deposita in $FB ■ $FC la locazione di start della stampa (prelevata, tramite puntatore, dalla tabela). Segue il prelievo del primo carattere che compone il fotogramma, quindi la stampa dello stesso. Si incrementa il puntatore ai caratteri ■, di conseguenza, il puntatore allo schermo. Terminata la prima riga di caratteri si prosegue con ■ seconda. Al termine, nel caso il fotogramma in questione fosse l'ultimo della serie, si cambia posizione e si azzera il contatore della pausa con il prossimo valore contenuto nella tabella, altrimenti... RTS. Come avrete notato la routine non ■

Foto 1

Foto 2

preoccupa di ripristinare il precedente contenuto dello schermo e ciò in quanto tutte le stampe devono essere effettuate in «aree libere». Vediamo ora la routine di commutazione delle scritte. In verità si tratta di commutazione tra due screen, che sono stati precedentemente memorizzati in due tabelle. La routine in LM è estremamente semplice e consiste in un semplice trasferimento dalla tabella allo schermo video, sotto impostazione della locazione di start di prelievo dati (schermata 1 o 2). Anche questa sub-routine viene utilizzata dalla routine IRQ di gestione presentazione. La routine di scrolling parallattico è composta da due sub che si occupano dello scrolling grosso e da una routine funzionante in IRQ, che tra l'altro si occupa anche della gestione delle altre sub (commutazione dei due screen

```
Routine di stampa delle stelle lampeggianti

LDX PUNTATORE TABELLA POSIZIONI E PAUSE
LDA TABELLA POSIZIONI E PAUSE,X
STA $FB
INX
LDA TABELLA POSIZIONI E PAUSE,X
STA $FC
LDX PUNTATORE FOTOGRAMMI
LDY #0
AST1:LDA TABELLA FOTOGRAMMI,X
STA ($FB),Y
INX
INY
CPY #3
BNE AST1
INC CONTATORE PASSATE
LDA CONTATORE PASSATE




CMP #2
BEQ AST2
LDX #0
LDA $FB
ADC #40
BCC AST1
INC $FC
JMP AST1
AST2:LDA #0
STA CONTATORE PASSATE
STX PUNTATORE FOTOGRAMMI
CPX ULTIMO CARATTERE ;(=NUMERO FOTOG. X 6)
BEQ AST3
RTS
AST3:INC PUNTATORE TABELLA POSIZIONI E PAUSE
INC PUNTATORE TABELLA POSIZIONI E PAUSE
LDX PUNTATORE TABELLA POSIZIONI E PAUSE
LDA TABELLA POSIZIONI E PAUSE,X
STA CONTATORE PAUSA TRA DUE STAMPE
INC PUNTATORE TABELLA POSIZIONI E PAUSE
LDA #0
STA PUNTATORE FOTOGRAMMI
RTS

Routine di stampa delle screen di presentazione

STA $FB
STX $FC
LDA LB START DATI TABELLA (1 o 2)
LDX HB START DATI TABELLA
STA $FD
STX $FE
LDY #0
LDX #0
PR1:LDA ($FD),Y
STA ($FB),Y
INY
CPY #40
BNE PR1
LDY #0
LDA $FB
ADC #40
BCS PR2
PR5:LDA $FD
ADC #40
BCS PR3
PR6:INX
CPX #NUMERO DELLE RIGHE
BEQ PR4
JMP PR1
PR2:INC $FC
JMP PR5
PR3:INC $FE
JMP PR6
PR4:RTS

Routine "SCR1P"

LDY #0
LDX PUNTATORE ALLE RIGHE DEL PRIMO STRATO




ER1:LDA TABELLA 1 PRIMA RIGA,X
STA PUNTATORE ALLA PRIMA RIGA DELLO SCHERMO,Y
LDA TABELLA 1 SECONDA RIGA,X
STA PUNTATORE ALLA SECONDA RIGA DELLO SCHERMO,Y
LDA TAB. 1 TERZA RIGA,X
STA PUNT. TERZA R. S.,Y
LDA TAB.1 QUARTA R.,X
STA PUNT. QUARTA R. S.,Y
INX
INY
CPY #40
BNE ER1
INC PUNTATORE ALLE RIGHE DEL PRIMO STRATO
RTS

Routine "SCR2P"

LDY #0
LDX PUNTATORE ALLE RIGHE DEL SECONDO STRATO
TR1:LDA TABELLA 2 PRIMA RIGA,X
STA PUNTATORE ALLA QUINTA RIGA DELLO SCHERMO,Y
LDA TABELLA 2 SECONDA RIGA,X
STA PUNTATORE ALLA SESTA RIGA DELLO SCHERMO,Y
INX
INY
CPY #40
BNE TR1
INC PUNTATORE ALLE RIGHE DEL SECONDO STRATO
RTS

Routine IRQ dalla presentazione

SEI
LDA #$7F
STA $DC0D
LDA LB ROUTINE PRIMA IRQ
LDX HB ROUTINE PRIMA IRQ
STA $0314
STX $0315
LDA POSIZIONE DEL RASTER PER LA PRIMA IRQ
STA $D012
LDA $D011
AND #$7F
STA $D011
LDA #1
```

e scintillio delle stelle). Analizziamo per prime le sub dello scrolling grosso. Approfittando del fatto che non è necessaria una grande varietà del passaggio possiamo utilizzare un sistema un po' diverso dal solito e questo rende la routine molto più semplice. In sostanza la mappa di caratteri che dovrà essere «scrollata» è memorizzata come una serie di 4 tabelle (per il primo stato) ognuna composta da 256 caratteri; dette tabelle corrispondono ciascuna ad una riga dello scrolling (infatti il primo stato è composto da 4 righe, mentre il secondo solo da 2). Ogni qualvolta occorrerà effettuare uno scrolling «grosso» si chiamerà la suboutine «SCR1P» (o la «SCR2P» per il secondo stato).

Da notare che non viene effettuato uno scrolling di quello che c'è sullo schermo e relativa aggiunta di caratteri

```
STA $D01A
LDA $D016
AND #$F8
ORA #6
STA POSIZIONE SCROLL 2 FINE IN X
ORA #1
STA POSIZIONE SCROLL 1 FINE IN X
CLI
RTS

-prima IRQ

LDA #$FF
STA $D019
LDA LB ROUTINE SECONDA IRQ




LDX HB ROUTINE SECONDA IRQ
STA $0314
STX $0315
LDA POSIZIONE DEL RASTER PER LA SECONDA IRQ
STA $D012
LDA $D011
AND #$7F
STA $D011
DEC POSIZIONE SCROLL 1 FINE IN X
LDA POSIZIONE SCROLL 1 FINE IN X
AND #07
BEQ HT1
LDA POSIZIONE SCROLL 1 FINE IN X
STA $D016
JMP $FEBC
HT1:LDA POS. SCR. 1 F. IN X
ORA #07
STA POS. SCR. 1 F. IN X
STA $D016
JSR ROUTINE "SCR1P"
JMP $FEBC

-seconda IRQ

LDA #$FF
STA $D019
LDA LB ROUTINE TERZA IRQ
LDX HB ROUTINE TERZA IRQ
STA $0314
STX $0315
LDA POSIZIONE DEL RASTER PER LA TERZA IRQ
STA $D012
LDA $D011
AND #$7F
STA $D011
DEC POSIZIONE SCROLL 2 FINE IN X
DEC POSIZIONE SCROLL 2 FINE IN X
LDA POSIZIONE SCROLL 2 FINE IN X
AND #07
BEQ JH1
LDA POSIZIONE SCROLL 2 FINE IN X
STA $D016
JMP $FEBC
JH1:LDA POSIZIONE SCROLL 2 FINE IN X
ORA #06
STA POSIZIONE SCROLL 2 FINE IN X
STA $D016
JSR ROUTINE "SCR2P"
JMP $FEBC

-terza IRQ

LDA #$FF
STA $D019
LDA FLAG PER COMUNICARE LA FINE DELLA PRESENTAZIONE
BNE SN1
JMP SN2
SN1:LDA LB ROUTINE IRQ GESTIONE JOY
LDX HB R. IRQ G. J.
STA $0314




STX $0315
LDA POSIZIONE DEL RASTER PER LA IRQ JOY
STA $D012
LDA $D011
AND #$7F
STA $D011
JMP ROUTINE MUSICALE
SN2:LDA LB ROUTINE PRIMA IRQ
LDX HB ROUTINE PRIMA IRQ
STA $0314
STX $0315
LDA POSIZIONE DEL RASTER PER LA PRIMA IRQ
STA $D012
LDA $D011
AND #$7F
STA $D011
LDA VALORE DI $D016 PER PRIMA PARTE (fisso)
STA $D016
JMP ROUTINE GESTIONE STELLE E CAMBIO SCREEN

Routine gestione stelle e cambio screen

LDA CONTATORE PAUSA TRA DUE STAMPE
BEQ CR1
DEC CONTATORE PAUSA TRA DUE STAMPE
JMP GESTIONE CAMBIO SCHERMO
CR1:JSR ROUTINE DI STAMPA DELLE STELLE LAMPEGGIANTI
-GESTIONE CAMBIO SCHERMO
DEC PAUSA TRA GLI SCREEN
BEQ CR2
JMP ROUTINE MUSICALE
CR2:LDA VALORE INIZIALE DELLA PAUSA TRA SCREEN
STA PAUSA TRA GLI SCREEN
LDA SCHERMO VISUALIZZATO
CMP #1
BEQ CR3
LDA LB START SCHERMO 2
LDX HB START SCHERMO 2
JSR ROUTINE DI STAMPA DELLE SCREEN DI PRESENTAZIONE
LDA #1
STA SCHERMO VISUALIZZATO
JMP ROUTINE MUSICALE
CR3:LDA LB START SCHERMO 1
LDX HB START SCHERMO 1
JSR ROUTINE DI STAMPA DELLE SCREEN DI PRESENTAZIONE
LDA #2
STA SCHERMO VISUALIZZATO
JMP ROUTINE MUSICALE
```

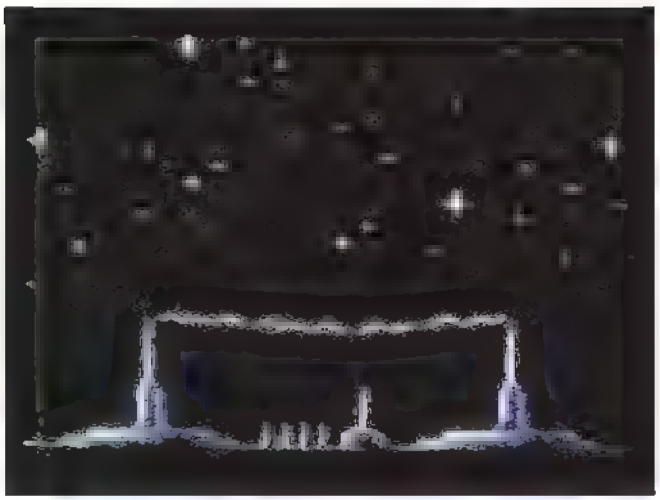


Foto 3

Foto 4

Megaposta

Caro Marco Pesce siamo due pseudo-programmatori di Rovigo, facenti parte del 1541 UserClub, i nostri nomi in codice sono Double P e Maxbunker; quest'ultimo, cioè colui che sta scrivendo in questo momento, si occupa da svariati anni di musica a tutti i livelli ■ in particolare di informatica musicale da quando il computer è entrato prepotentemente nel line-up domestico di molte famiglie italiane, prima cominciando con il glorioso «ever-green» C-64 per continuare poi con un Atari ST, vista l'esperienza in ambito MIDI accumulata col primo.

In particolare, passando dal C-64 all'Atari, l'utilizzo in MIDI del primo è andato scemando a causa della mancanza di software specifico; mi spiego meglio: di software MIDI per C-64 ne esiste una marea, sequencer, composer su partitura, editor per sintetizzatori, utility varie, ma manca totalmente l'anello di giunzione tra l'intramontabile SID 6581 (che giudichiamo uno tra i migliori integrati audio mai creati) ■ la MIDI, il modo dei sequencer professionali, delle master keyboard, dei sintetizzatori professionali ■ dei campionatori. Nessuno ha mai pensato di realizzare un programma di editor del SID con la possibilità di controllare l'input delle note da una tastie-■ MIDI e non solo dalla misera tastiera alfanumerica del C-64, non certamente nata per scopi musicali.

Veniamo al sodo: qui a Rovigo stiamo progettando di realizzare un editor musicale (o adattare un editor già esistente) comandabile da una tastiera MIDI o, più estesamente, da un sequencer professionale che implementi possibilità di modifica ben più sofisticate di un qualsiasi editor realizzato per C-64 e che permetta anche l'input in tempo reale dei dati, cosa negata alla maggior parte degli editor che non va oltre l'input step by step sotto forma di un poco musicale linguaggio di programmazione.

Siamo sicuri che gente come Rob Hubbard, Martin Walker, David Whittacker avranno già realizzato qualcosa di simile, visti i risultati delle loro musiche (soprattutto negli assoli) che non sembrano per niente lunghe liste di numeri e parole chiave digitati sulla tastiera alfanumerica del C-64...

Caro Marco, noi ti abbiamo contattato per chiedere la tua collaborazione nella realizzazione di questo ambizioso progetto...

Massimiliano Pigozzi ■
Pierpaolo Bergamo, Rovigo

Cari amici di Rovigo, innanzi tutto complimenti per il demo tape allegato al pacco. Per quanto riguarda il vostro progetto credo che accontentarvi in tutto e per tutto sarà un po' difficile, in quanto sono del parere che l'interesse per questo tipo di programmi sia davvero scemato anche tra i lettori di MC (se mai c'è stato). Tuttavia era nelle mie intenzioni quella di riprogettare un nuovo ■ più flessibile editor musicale per il C-64, che non abbia i limiti del precedente, quale l'intricato sistema di input in tempo reale da tastiera o il ridotto set di strumenti utilizzabili; tra l'altro ■ parte Basic rallentava in modo eccessivo la gestione delle opzioni ■ in una prossima release vorrei eliminarla. Non è escluso comunque che, nelle prossime puntate della Scuola di Videogame sull'argomento, vi suggerisca qualche dritta per il vostro progetto...

a destra (scrolling verso sinistra), ma lo schermo viene ogni volta ristampato! Per fare quanto detto basta stampare 40 caratteri (anche se sullo schermo se ne vedono solo 38) per ogni riga, prelevandoli dalle relative tabelle, e cambiare il carattere iniziale di volta in volta... più facile ■ farsi che a dirsi. Ma andiamo ad esaminarci la routine di gestione delle IRQ. Lo schermo è diviso in tre sezioni, ognuna avente un diverso valore della regolazione fine in X (primi 3 bit della locazione $D016); la prima parte ■ quella superiore ed è fissa, perché contiene le stelle e le scritte. La seconda parte ■ quella che si occupa dello scrolling del primo strato ed è composta da 4 righe. Infine la terza parte si occupa dello scrolling del secondo strato, che avviene a doppia velocità rispetto al primo.

Dopo la solita procedura iniziale di spostamento del vettore IRQ ■ relativo tipo di interruzione (legata ■ pannello elettronico) si imposta la prima riga alla quale dovrà verificarsi la relativa prima IRQ. Al verificarsi di detta IRQ si effettua l'ovvio spostamento del vettore e abbinata nuova posizione del raster, quindi si imposta il valore dello scroll fine relativo al questa sezione ■ si effettua il suo decremento, con eventuale scrolling grosso. La stessa procedura, ma con valori diversi, si effettua nella seconda IRQ. La terza IRQ non comprende scrolling; comprende invece una sezione che si occupa dell'intrerruzione del ciclo di IRQ nel caso il flag fornito dal «main-program» sia impostato. C'è inoltre la gestione delle stelle e dello scambio tra gli schermi, che si effettuano basandosi su dei contatori e sulle subroutine illustrate in precedenza. Per questa puntata è tutto... buon Natale.

di Marco Pesce

Scuola di videogame

Il discorso che andiamo ad affrontare in questa puntata si riferisce al progetto di una routine per il movimento dei «nemici» in uno «spara e fuggi». Alcune considerazioni sono valide in generale, ma effettueremo una distinzione fra tre categorie che illustreremo appresso

Cominciamo dall'ipotesi più banale, ovvero dal caso semplice di un determinato «set» di nemici (vi ricordo che sotto questo nome consideriamo ogni tipo di ostacolo che il giocatore incontrerà durante il suo tragitto), che agiscono secondo schemi prefissati e con un ordine prestabilito. In tal caso la routine è costituita da un insieme di algoritmi per il movimento (ad esempio una decina) e da una tabella che determina i tempi e le modalità di attacco dei nemici. Facciamo un esempio: un gioco «spaziale» a scrolling orizzontale verso sinistra (vedi foto 1) dispone di una tabella che contiene una serie di eventi, collegati allo scrolling, ovvero, ogni qualvolta si giunge ad una posizione della coordinata X contenuta anche nella suddetta tabella, si attiva una nuova routine di movimento e quindi dei nuovi sprite con relative animazioni, che in genere è meglio definire con tabelle di contorno. Supponiamo quindi (foto 2) che l'attuale posizione dello scrolling faccia attivare l'ingresso in campo di sette nuovi nemici (in foto solo 4); evidentemente nel caso del Commodore 64 (a meno di «moltiplicazioni») il nuovo ingresso deve coincidere con una «uscita», pena «sfarfallamenti» indesiderati. Andiamo a considerare alcuni aspetti della routine. I problemi da risolvere sono:

— inizializzazione;
— movimento;
— esclusione dei «morti»;
— uscita;
— gestione ordinata degli sprite disponibili.

Nel caso del gioco delle foto 1 e 2 (che possiamo far rientrare nella «categoria 1») le cose sono notevolmente semplificate (in genere) perché il fondale è ininfluente, o meglio non fa da vincolo ■ quindi le routine di movimento sono semplici sequenze di posizioni da assumere (o funzioni matematiche che restituiscono dette posizioni).

Il problema della inizializzazione è strettamente collegato con la gestione ordinata degli sprite, che tuttavia, per la categoria 1, è da considerarsi semplice. Sostanzialmente si tratta di eliminare gli sprite che sono in campo se servono per la «formazione» in ingresso (ad esempio con una esplosione), ma in genere il problema è ancora più semplice, in quanto possiamo fare in modo che sullo schermo non compaiano nuovi sprite se quelli disponibili sono già tutti utilizzati. Dal momento che non dobbiamo preoccuparci di questo problema, possiamo dedicarci alla struttura della tabella; essa deve comprendere informazioni sulla posizione di partenza e sulla routine da attivare per il successivo movimento (consideriamo la tabella costituita da informazioni relative ad un solo nemico per volta), più la posizione di controllo dello scrolling. Se vogliamo ottenere l'effetto della foto 2 (che purtroppo è statica!) ognuno dei sette nemici deve essere attivato ad una posizione dello scrolling molto vicina a quella del precedente ed utilizzando la stessa routine di movimento (che quindi deve essere «multi-utente»), facendo attenzione ■ calcolare nel frattempo anche la nuova posizione di partenza (visto che lo schermo si è spostato) se vogliamo che i nemici «entrino» tutti dallo stesso punto. Se vogliamo che un certo nemico sia formato da più sprite messi assieme basta attivare il numero di sprite necessari (ad esempio 4) nello stesso momento, quindi alla stessa posizione dello scrolling, ■ posizionarli nel modo giusto, ovvero secondo l'effetto che vogliamo ottenere. Quest'ultima possibilità ci impone una regola nel progetto delle routine di movimento; esse infatti dovranno gestire un massimo di 7 sprite (1 è del giocatore... e non abbiamo contato i «colpi») e quindi 14 coordinate indipendenti, che tuttavia seguiranno sempre lo stesso schema, in modo da fornire lo stesso spostamento (vi faccio presente che le routine non devono fornire coordinate precise, ma solo le direzioni di spostamento) se vengono attivati più sprite contemporaneamente. Ciò permette anche di ottenere l'effetto «serpente», se gli sprite sono parzialmente sovrapposti e sono attivati a breve distanza. Non è da escludere ad ogni modo una routine che fornisce spostamenti casuali, che ovviamente è da utilizzarsi solo nel caso di sprite singoli.

Facciamo uno schema di riepilogo per la tabella:

— posizione dello scrolling;
— posizione di start (x, y);
— routine da utilizzare;
— animazione da utilizzare (con sotto-

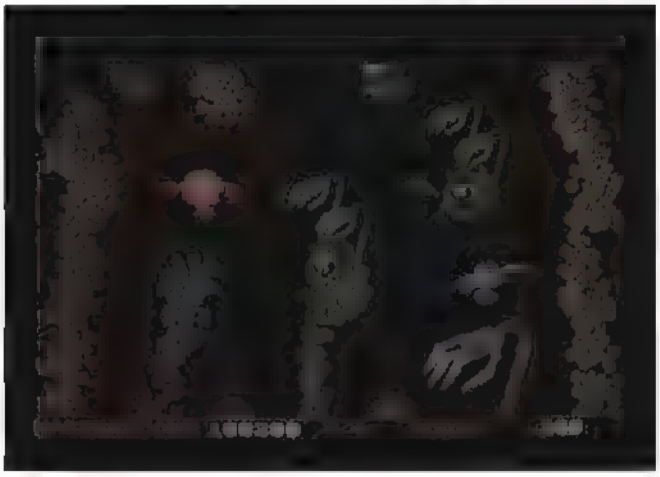


Foto 1

Foto 2

tabella);
ripetuta per ogni sprite-nemico.

Veniamo alle routine di movimento e supponiamo che il loro sistema di definizione degli spostamenti sia costituito da un insieme di tabelle (quindi niente funzioni matematiche). Una volta attivate esse dovranno fornire una direzione di spostamento per ogni ciclo raster ■ per ogni sprite che la utilizza. L'inizializzazione consiste nell'azzeramento del puntatore alla tabella. In conseguenza dell'attivazione esse restituiranno (per il relativo sprite), in appositi registri, la direzione di spostamento, tra le otto + una (spostamento nullo) disponibili. Tengo ■ precisarvi che gli sprite subiscono anche un decremento della loro coordinata X per ogni spostamento dello scrolling. Volendo fare le cose un po' più sofisticate possiamo anche indicare, oltre alla direzione, la quantità di spostamento, al fine di ottenere effetti di accelerazione e viceversa. Le tabelle di spostamento possono essere lunghe a piacere ■ magari cicliche, per ottenere ad esempio, una continua rotazione intorno ad un punto, con possibilità di definire la posizione di restart per il ciclo, in modo da separare la fase di ingresso ■ quella di loop (l'ingresso dall'alto ■ la rotazione ciclica si ottengono definendo la posizione di restart coincidente con l'inizio della fase circolare).

Supponiamo quindi di avere 10 (diciamo subito che in genere sono poche) di queste routine, ognuna con una determinata traiettoria. La loro struttura è del tipo di quella seguente:
— attendi l'attivazione;
— se attivata per la prima volta «reset»;
— deposita lo spostamento per ogni sprite attivo nei relativi registri;
— incrementa i puntatori (attivi) alla tabella;
— «reset» in caso di «end-tabella».

Esistono poi 7 registri per indicare lo spostamento e la quantità dello stesso (ad esempio «nord-3 spazi»).

È evidente che le traiettorie dovranno essere ben collaudate, ovvero non dovranno provocare strani movimenti, ma questi sono problemi scontati. Associate alle routine di movimento abbiamo le routine di animazione ■ controllo dell'uscita dallo schermo o «morte». In sostanza esse devono prelevare i registri di spostamento solo nel caso il nemico sia ancora vivo ■ devono effettuare l'animazione che per semplicità supponiamo ciclica (ad esempio con 4 fotogrammi), ma che volendo potrebbe anche essere in funzione della direzione (visto che le informazioni ci sono). In caso di uscita, questa routine deve disabilitare lo spostamento ■ rendere disponibile lo sprite. In caso di «morte», si deve attivare la sequenza di esplosione e, una volta terminata, rendere disponibile lo sprite.

Sintetizziamo tutto il discorso con un esempio chiarificatore (speriamo). Il nostro fondale ■ in scrolling e la nostra astronave vola liberamente. Arrivati ad una determinata posizione la routine di controllo della tabella degli eventi verifica una attivazione ■ quindi imposta la routine indicata nei dati, la posizione di partenza e l'animazione da utilizzare. Fatto questo la palla passa alle altre routine; innanzi tutto viene scelto uno spirite tra quelli a disposizione, poi la routine di spostamento effettua il suo dovere e segue la routine di animazione e controllo «morte», che utilizza la tabella indicatagli dalla inizializzazione. Chi stabilisce la morte è la routine delle collisioni che in questa sede non trattiamo. Di contorno ci sono ancora subroutine che si occupano di gestire il chip video, quindi di trasformare le informazioni in modo da inserirle nei registri opportuni.

Abbiamo quindi concluso il caso della «categoria 1». In conclusione di questa parte vorrei darvi un consiglio, valido in generale per qualsiasi progetto; se dovete realizzare insiemi di dati complessi e suscettibili ad errori di compilazione, vi conviene «costruirvi» un editor specifico; il tempo speso vi verrà largamente ricompensato in seguito.

I «più attenti» avranno capito che le restanti foto ■ riferiscono alle categorie 2 ■ 3 (rispettivamente, foto 3 ■ 4). La categoria 2 è quella degli spara e fuggi a scorrimento verticale (ma volendo pos-

Foto 3

Foto 4

siamo farli anche a scorrimento orizzontale) con vista dall'alto e con fondale interattivo. Quest'ultima caratteristica ci complica notevolmente le cose, in quanto la traiettoria deve essere realizzata con un algoritmo e non con una semplice sequenza di spostamenti.

Ciò modifica solo la routine che determina la traiettoria, ma non il sistema di attivazione; al limite possiamo effettuare anche un ibrido tra i due metodi di definizione della traiettoria, in quanto non è escluso che vi siano nemici «volanti» e che quindi non vengono vincolati dal terreno. Anche la categoria 3 (vista laterale con fondale interattivo) utilizza lo stesso metodo, ma gli algoritmi sono notevolmente differenti, visto che in questi giochi si tratta soprattutto di «saltare» eventuali «buche» o «muri», mentre per la categoria 2 basta cambiare direzione se si incontra un «muro». Tuttavia in entrambi i casi occorrono routine di controllo del fondale che ci costringono a realizzare tabelle per specificare dove si può passare e dove ci deve fermare.

Una totale rivoluzione la possiamo effettuare se decidiamo di inserire nemici «intelligenti». Dovremo realizzare algoritmi capaci di individuare le mosse dell'avversario (giocatore) e quindi di controbattere, magari effettuando «finte». Un esempio lampante sono i giochi di lotta («International Karate»). In aggiunta agli algoritmi di movimento vincolato vanno quindi costruite routine di ricerca della posizione (sia dell'avversario che di eventuali «proiettili») e scelta della mossa da eseguire (quest'ultima al limite potrebbe anche essere fatta casualmente). È chiaro che il nemico non deve essere invincibile. L'ingresso in campo di un nemico di questo genere potrebbe essere predeterminato come nei casi precedenti o, volendo, anche casuale (senza esagerare). Se si riesce ad effettuare un giusto miscuglio delle tecniche descritte il risultato è senza dubbio più soddisfacente (possiamo, in fase di gioco, differenziare i nemici in stupidi e ingegnosi e tutto diventa più divertente). Adesso però devo salutarvi... a presto.

Megaposta

«Carissimo Marco Pesce, sono un ragazzo di 20 anni e frequento il secondo anno di ingegneria. Ritengo di averti già sommerso di informazioni inutili quindi passo al dunque della mia missiva. La prima cosa da dire è che non sono un assiduo lettore di questa rivista, ma interessandomi di videogame sono stato colpito dalla tua iniziativa (un po' in ritardo forse, ma purtroppo l'ultima volta che ho comprato MC ancora non c'era traccia del tuo operato!). La cosa che mi piace di più è il tipo di V.G. scelto, che non rientra assolutamente tra i più diffusi (almeno stando a quanto ho capito). Io sono dell'idea che un gioco al computer deve essere considerato un'opera d'arte e in quanto tale deve offrire qualcosa di inconsueto per essere degno di nota. E sembra proprio che il tuo caso sia da considerarsi perlomeno eccentrico. Ecco, questa era la cosa più importante che volevo dirti. C'è anche un'altra cosa che voglio dirti: quando pensi che sarà pronto?...».

Massimiliano Vetriani, Roma

Finalmente qualcuno che si preoccupa del megagame... Vogliamo fare una previsione? Penso proprio che per il numero di marzo potremo cominciare le nostre battaglie intergalattiche... grazie dei complimenti.

Caro maestro (Marco Pesce), è da molto che attendo una tua svolta per la scuola di videogame, nel senso che ormai sarebbe ora di passare al computer del momento (il piccolo grande Amiga); mi sembra che anche tu avevi accennato a una cosa del genere. Credimi, a sentire i pareri di molti sembra proprio che riscuoteresti un successone...».

Luca Cesaroni, Milano

A parte il fatto che non ho capito di chi sono questi «pareri», comunque mi trovo in perfetta armonia con te. In effetti hai ragione, del resto è molto che mi occupo di questo computer (in privato) senza mai aver accennato nulla su queste pagine... ancora poche battute per il 64 e poi vedremo cosa si può fare.

di Marco Pesce

Un videogioco tutto nostro

Siamo giunti al nostro penultimo resoconto sullo stato del nostro videogioco. Il prossimo mese vi dovrei presentare il programma finale e allora tireremo le nostre belle conclusioni. Per questa puntata ci occuperemo di osservare, come al solito, la situazione raggiunta

In quest'ultimo mese ho realizzato gran parte del lavoro «concreto», vale a dire programma già testato e funzionante e totale della grafica, più alcuni esperimenti musicali. Chiaramente il programma non è ancora finito, altrimenti potevamo concludere con questa puntata, e manca la colonna sonora (ovvero c'è il programma per realizzarla ma non il musicista... ovvero c'è il musicista ma non l'idea per la musica... ovvero c'è l'idea ma non il tempo per realizzarla... fino ad oggi). Ci incamminiamo quindi in una serie di spiegazioni che, spero, serviranno per far rivivere anche a voi le mie stesse emozioni riguardo questo progetto.

Al termine della scorsa puntata mi ritrovavo con alcune schermate realizzate tramite Kwik paint (un prg per disegnare sul Commodore 64) e un grande schema della struttura del gioco, più una manciata di routine per la presentazione tra l'altro neanche testate «bene». L'unica parte del programma che era stata effettivamente trasferita sul computer e vista funzionare era la routine musicale. Le schermate erano relative alla presentazione, ma in passato ne avevo realizzate diverse come modelli per il gioco effettivo e molto spesso erao state scartate perché il gioco e era modificato lungo il cammino. Nonostante ciò mi era piaciuta l'idea di alcune schermate pubblicate sul numero di gennaio dell'88 (più di un anno fa!). Ovviamente non andavano bene per il Megagame attuale, ma il loro stile mi convinceva. Vada per lo stile, mi metto quindi all'opera per adattarle ai pannelli rimodernati (vedi numero di novembre '89). E in queste pagine potete infatti ammirare i nuovi pannelli made in C-64.

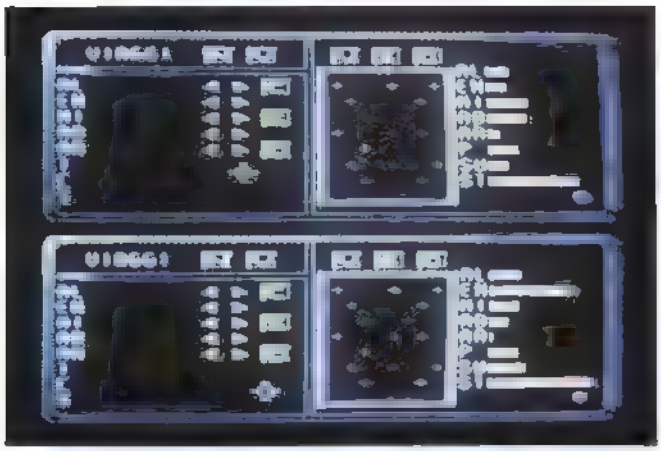
Terminate le schermate con il Kwik paint occorreva renderle utilizzabili, quindi trasformarle in set di caratteri. Questo era solo il primo passo da fare in quanto, una volta mescolate nei set (tre) di caratteri occorrevano delle mappe che avrebbero ricostruito le schermate secondo i nostri comandi. La soluzione era più semplice del previsto in quanto mi occorreva un programma in grado di «costruire» queste mappe e a suo tempo avevo realizzato qualcosa del genere per il videogame Psicoman (rimasto incompleto...), bastava modificare «un po'» di cosette. Mi ritrovo alla fine con un bel tool per la creazione delle mie mappe, ma mi servivano ancora i set di caratteri. Il primo dei tre riguardava la grafica della presentazione (stelle, montagne, scritte e set alfabetico) che ha dato fortunatamente pochissimi problemi (solo un carattere non coincideva con una scritta). Il procedimento di trasformazione consisteva nel «ritagliare» parti di screen effettivamente occupate dal disegno e comprimerle in una ristretta area di schermo in hi res; fatto questo bastava salvare la schermata e poi prelevare solo la parte occupata (l'inizio della screen... i primi 2 kbyte, coincidenti appunto con l'ampiezza di un set di caratteri). Come forse non sapete, o non ve ne siete mai resi conto, le screen in hi-res (sia multi che mono color) sono memorizzate in ram con lo stesso ordine della mappa caratteri, ovvero non c'è bisogno di conversione tra i due formati (a differenza degli sprite), quindi il «gioco» era già fatto senza ulteriore sforzo, tutto stava a spostare la parte di schermata che ci interessava nell'area ram da noi dedicata al (primo) set di caratteri. Chiaramente la procedura dei «tagli» illustrata prima richiede una certa attenzione in quanto dobbiamo far coincidere esattamente in un'area di 8 per 8 pixel (allineata con 8 byte) ogni carattere, sempre che questo debba essere utilizzato da solo come nel caso dei caratteri facenti parte del set alfabetico; nel caso di

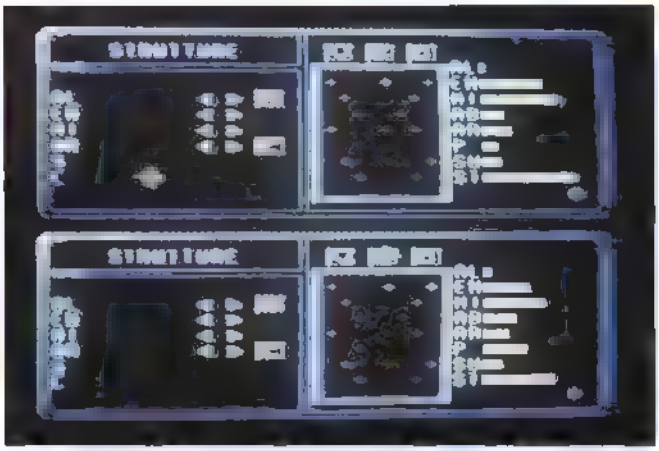
gruppi di caratteri che poi saranno stampati uno accanto all'altro non ci sono particolari problemi di sconfinamento, almeno per quelli interni al gruppo. Tutto questo per farvi capire che mi sono ritrovato più volte a dover riaggiustare la schermata «tagliuzzata» perché un determinato carattere di 8 per 8 era diviso in due, oppure non si incastrava» bene nel mosaico (specialmente per quanto riguarda il set dei pannelli). Il Kwik paint utilizza solo il modo multicolor e infatti tutto il megagame è in multicolor, caratteri alfabetici compresi. Per gli sprite, che poi si riducono a soli cursori animati, ho usato un programma di conversione già utilizzato per lo stesso Psicoman. Tanto per la cronaca il secondo set è quello dei pannelli e il terzo quello della schermata del «famoso» fine-giorno. Nel caso qualcuno di voi volesse impossessarsi del mio tool per la creazione di mappe caratteri vi illustro alcune sue caratteristiche. Sullo schermo è visualizzata una bella fetta di quella che è la nostra pagina video, più l'intero set di caratteri (sia ridefiniti che normali, a scelta). Si costruisce tramite joystick con il cursore che si trasforma nel carattere da noi scelto tra i disponibili. Possono essere definiti fino a 9 «pennelli», ovvero agglomerati di caratteri, prelevati dallo screen stesso o dal set, richiamabili in qualunque momento (con notevole risparmio di tempo). Tramite altre opzioni è possibile richiamare screen memorizzate in ram secondo un ordine scelto da noi (in quanto è possibile riposizionare liberamente il puntatore che stabilisce la sezione di ram) e ovviamente salvarle (sempre in ram). C'è anche la gestione dei file su disco (save e load). Se vi interessa fatemelo sapere che ne riparliamo. Torniamo al Megagame. Siamo giunti al momento in cui mi ritrovo finalmente con le mie mappe e i miei set di caratteri, più gli sprite. Stabilisco la loro posizione in ram e li salvo in un unico blocco, al fine di poterli ricaricare ogni volta che effettuerò un test sulle routine Assembler che mi accingo a realizzare.

Le prime routine che trasferisco sul Commodore, servendomi del solito Assembler, sono quelle della presentazione. Mi accorgo che devo fare alcune modifiche perché parte di esse possono essere utilizzate come sub anche per il resto del programma e perché ci sono alcune imperfezioni. Dopo non poche lotte con la routine dello scrolling parallattico, che non voleva assolutamente essere fluido (ma l'ho spuntata io) la presentazione è completa e la mostro orgogliosamente a mio fratello. Mancano ancora i nomi da inserire nella lista degli «hanno partecipato». Ma la routine che più mi rende felice è quella della stampa di fette di ram di qualunque dimensione, affiancata poi da quella di riempimento con valore fisso (per le cancellazioni). L'unico inconveniente è che per funzionare a volte richiedono (se la fetta è troppo grande) più di un ciclo raster (1/50 di secondo), ma ciò non crea alcun problema (almeno fino ad oggi!).

Quello di cui ho bisogno a questo punto è un elenco dettagliato delle loca-

Sarei tentato a lasciarla così





zoni di partenza di tutte le schermate contenute nel blocco dati della mappa realizzata in precedenza. Mi realizzo una modifica al programma di editing delle mappe in modo che sia lui a fornirmi le mie informazioni e il problema non sussiste più. In questo elenco occorre aggiungere anche le locazioni dello schermo che andranno ad occupare, una volta stampate le relative screen, i tasti dei pannelli, le scritte seguite da informazioni a barra, la posizione dei pianeti, più gli indicatori vari. Anche se il tutto è diverso nel caso dei pannelli del secondo giocatore, basta aggiungere un delta di 520 posizioni ed i dati ottenuti risultano perfettamente coincidenti.

Consideratelo pure un capriccio da programmatore, ma le prossime routine che mi propongo di realizzare sono quelle di gestione del joystick, della stampa dei pianeti sulle mappe e della stampa degli indicatori a barra. Butto giù degli schizzi e passo subito alla scrittura delle routine in Assembler; in effetti l'impresa mi era sembrata più complessa di quanto effettivamente poteva essere dopo alcune semplici modifiche di criterio. Durante l'assemblaggio di tanto in tanto carico il blocco di dati con la grafica per assicurarmi che tutto vada per il meglio, ma il più delle volte è solo perché sono curioso di vedere il risultato di ogni minima modifica (e la cosa mi comporta un bel po' di caricamenti-attese-perditempo). Tanto per avere in mano qualcosa di concreto mi faccio un paio di demo con il risultato di quanto ho fatto fino a quel momento.

Andiamo ad esaminarci queste routine. Cominciamo con quella di gestione del joystick. Come abbiamo detto il cursore-puntatore è uno sprite (animato) quindi le sue coordinate non sono direttamente utilizzabili per essere confrontate con la posizione sul video dei vari gadget, che sono invece in formato 40×25. Tuttavia è sufficiente dividere per 8 sia la coordinata X che la Y dello sprite e otteniamo una coppia compatibile con lo schermo caratteri. Il carattere in alto a sinistra (locazione ram-video 1024) corrisponde alle coordinate 0,0 (X,Y) e il secondo (loc. ram 1025) alle coordinate 1,0 e così via. C'è ancora un piccolo problema da risolvere, in quanto quando lo sprite occupa le coordinate 0,0 non coincide (in termini di sovrapposizione fisica) al primo carattere e per questo occorre aggiungere una delta di 22 pixel in X e 50 in Y (considerando anche la forma dello sprite). Tutto ciò che riguarda questa routine funziona in IRQ video.

Passiamo alla routine di gestione della stampa degli indicatori a barra. Questi indicatori sono costituiti da una barra di pixel in orizzontale (avente una ampiezza verticale di 4 pixel), tanto più ampia quanto maggiore è la grandezza che con essa si intende rappresentare. La barra è quindi costituita da un insieme di caratteri ridefiniti in multicolor. La sua max ampiezza orizzontale è sempre in 54 pixel (quindi 8 caratteri). Se osservate le foto relative ai pannelli potete ritrovare una serie di barre nella destra della mappa. Il mini set di caratteri abbinato alla routine è costituito da 8 «pezzi», aventi ognuno una striscetta di quattro pixel verticali in più rispetto al precedente. Per realizzare una barra di 10 pixel occorrono 2 caratteri, uno di 8 pixel (pieno) e l'altro di 2, messi uno accanto all'altro. La routine trasforma un qualsiasi numero compreso tra 0 e 63 in una corrispondente barra, posizionata sullo schermo a nostro piacere. Il suo funzionamento è abbastanza semplice. È sufficiente avere un puntatore alla locazione video di partenza e il numero da trasformare. Del numero si prendono i tre bit meno significativi e si depositano in un registro libero; questi tre bit indicano quale degli 8 caratteri occorre stampare dopo la serie di caratteri «da 8 pixel». Quest'ultima serie di caratteri viene stabilita tramite una divisione per 8 del numero stesso. Si effettua quindi un ciclo di stampa servendosi del puntatore video e incrementandolo di volta in volta. Il ciclo ovviamente dura tanto quanto è il risultato della divisione (nel caso dell'esempio, un solo ciclo). Al termine del ciclo si stampa il carattere stabilito dai tre bit meno significativi, ricavati in precedenza, nella posizione video indicata dal puntatore attuale.

Veniamo alla routine di stampa dei pianeti. La mappa è di 8×7 caratteri, ciò vuol dire che è compresa in un range di 6×40+8=240 locazioni, quindi è sufficiente l'indirizzamento indicizzato per ricoprire l'intera area. La loro posizione effettiva sullo schermo è determinata da una locazione di base (diversa, ovviamente per le due mappe, dei due giocatori) più una delta (registro X) che va da 0 a 240, contenuto in una opportuna tabella di 16 elementi. La loro forma dipende dal loro stato (normalità, pericolo o guerra) e per stabilirla occorre ricercare i flag che lo indicano nella tabella dati del pianeta. Anche il colore di stampa dipende da questi dati (mio-tuo, attivo-disattivo).

E adesso occupiamoci della routine che utilizzano come «schiavi» le sub finora descritte, ovvero la routine di gestione dei tasti e conseguente attivazione delle funzioni corrispondenti, la routine di gestione dei pianeti e la routine di gestione della stampa dei pannelli. In parte queste routine si fondono e in parte svolgono funzioni a sé stanti. Ad esempio, la routine dei tasti deve controllare che il joystick coincida con uno dei tasti contenuti nei pannelli ovvero che le coordinate di questi, contenute nelle tabelle già descritte, coincidano con quelle del puntatore, anche per la gestione dei pianeti occorre un confronto di coordinate per stabilire quale è quello attivo, quindi a quale devono riferirsi le caratteristiche stampate con le barre. La routine dei tasti non fa altro che settare un flag quando verifica la coincidenza; sarà poi il resto del programma ad occuparsi della funzione relativa. Quando si attiva un nuovo pianeta la mappa viene ristampata per aggiornare «l'illuminazione» dei pianeti, ovvero per spegnere quello vecchio e accendere quello nuovo (vi ricordo che un pianeta è acceso quando passa dal suo colore base, blu o rosso, al rispettivo colore più luminoso, ciano o giallo), quindi viene ristampato l'elenco di basse sulla destra, prelevando i dati dalla tabella del pianeta attivato.

La stampa dei pannelli si effettua basandosi sulle seguenti informazioni:
— giocatore utilizzatore,
— tipo di pannello,
— tipo di sottopannello,
quindi la prima informazione determina dove stampare il pannello, la seconda qual è l'opzione e la terza qual è la sotto opzione. Si procede quindi con la stampa, solo se i pannelli attivi sono cambiati rispetto al ciclo programma precedente.

Megaposta

Caro Marco Pesce,
finalmente tra tutti i lettori della rivista MC c'è anche una ragazza che s'interessa di videogiochi. Compro regolarmente la rivista e leggo volentieri anche la rubrica di Francesco Carlà, pur trovando spesso discutibili i suoi giudizi.

Ritengo la tua iniziativa veramente meritevole dello spazio che ha occupato finora nella rivista (se non vado errata, sono più di 2 anni che il Megagame «procede»). Mi rendo conto che è stata un'impresa difficile, ma spero che tu possa persistere perché trovo veramente interessante quello che scrivi e penso di non essere la sola, e spero che tu possa continuare ancora per molto tempo. Spero che questa lettera ti sia di conforto in qualche modo, ma non credo che tu ne abbia bisogno vista la tua tenacia. Aspetto con impazienza il prossimo numero!

A presto

Barbara Tosti, Roma

Erano due anni che aspettavo una lettera così. A quanto pare l'attesa è stata più che ricompensata! Cara Barbara, spero proprio che questa non sarà la tua ultima lettera. prossimamente fammi sapere se sei interessata a qualche argomento particolare.

Ciao.

di Marco Pesce

Scuola di videogame
«C-64 vs Amiga»

In questa puntata dovevamo concludere il lungo cammino del megagame con la presentazione del gioco definitivo, abbinata a una sorta di manuale di istruzioni. Sfortunatamente ci sono stati alcuni ritardi e quindi per questo mese dobbiamo accontentarci della scuola di videogame, che ospita una specie di confronto tra le possibilità del nostro piccolo e quelle della sorella Amiga; il tutto con la manifesta intenzione di cominciare a trattare anche quest'ultima come oggetto di studi per la nostra scuola. Inoltre vi presenterò un piccolo editor di videogiochi per Amiga

Cominciamo con una descrizione sommaria, specifica per i nostri interessi, delle caratteristiche di Amiga; processore a 16 bit con clock a 7 MHz, RAM da 512 kbyte (di base), risoluzione video ottimale per videogiochi di 320×200 pixel in 16 o 32 colori (ne ha anche molte altre che però... non ci interessano), chip sonoro a 4 voci a campionamento ad 8 bit con suddivisione dei canali stereofonica, chip BLITTER per spostamenti veloci di zone RAM (soprattutto grafica), chip COPPER per la gestione del display video con elevata precisione di intervento sul pennello elettronico, 8 sprite hardware con 4 colori ciascuno (oppure 4 con 16 colori) con risoluzione di 16 X illimitati pixel, disk drive incorporato veloce (tranne che con le directory...) da 880 kbyte e due porte joystick. Caratteristiche così allettanti non potevano lasciare inerte la massa di programmatori di videogame, che infatti in questi ultimi anni ha tirato fuori da questo computer giochi di qualità sempre più elevata (anche se ogni tanto sembra di tornare ai ... vecchi tempi, quando tutto si muoveva con «terribili scatti»).

Ed ora uno sguardo al C-64; processore ad 8 bit con clock di 1 MHz, RAM da 64 kbyte, risoluzione video ottimale per videogiochi di 160×200 pixel con 4 colori, chip sonoro a 3 voci monofonico ma molto versatile, interruzioni hardware abbinate al pennello elettronico per una discreta gestione del display, 8 sprite hardware con 4 colori e risoluzione di 12×21 pixel, disk drive (esterno) non molto veloce da 170 kbyte, due porte joystick.

Nonostante la netta superiorità di Amiga su C-64 quest'ultimo non si è ancora deciso a cedergli definitivamente il passo. La spiegazione di ciò è ovviamente nel fatto che il C-64 esiste da tantissimo tempo ed ha avuto una diffusione incredibile. Inoltre lo si è studiato talmente a fondo da raggirare gran parte di quelle che inizialmente erano limitazioni, ma questa è un po' una conseguenza della prima motivazione. Il C-64 è quindi diventato una specie di standard per i videogiochi ed è quindi ancora oggetto di interesse di molte software house. L'Amiga esiste già da un bel po' di tempo ormai ed è una fortuna che non abbia fatto la fine di alcuni suoi colleghi, aventi caratteristiche hardware alla pari se non superiori (senza fare nomi). Ne è passato di tempo da quando sono stati presentati i primi giochini stile C-64, che a me personalmente mi mandavano in bestia; tutta quella potenza sprecata e nessuno se ne rendeva conto, o meglio nessuno voleva. Per vostra fortuna non voglio tirare di nuovo in ballo la storia di Amiga, che ormai risulterebbe decisamente obsoleta.

Un editor di videogame

Con la scusa del confronto vi presenterò una sorta di editor di videogiochi per Amiga, realizzato dal sottoscritto per produrre un giochino molto divertente (secondo il mio parere), che può essere utilizzato per la «costruzione» di qualunque spara e fuggi terrestre (tipo G'n'G... tanto per cambiare) a scorrimento orizzontale.

L'idea di partenza è quindi un giochino avente protagonista un personaggio umano (o simili...) che deve a tutti i costi avanzare lungo un territorio pieno zeppo di mostri e pericoli di ogni tipo, tanto intenzionati a ridurlo senza un briciolo di energia. Un limite di tempo costringe il nostro eroe a non soffermarsi troppo ad osservare il panorama. Tanti tanti punti se si ammazzano i cattivoni.

La struttura base di un gioco-programma del genere è la stessa qualunque sia il computer sul quale viene realizzato: le differenze si riscontrano nel lato estetico e nel modo di gestire le risorse hardware della macchina. Per C-64 esiste una serie incredibile di videogiochi di questo tipo, e a suo tempo anch'io ne realizzai uno, che però non ho mai completato. Per Amiga la situazione si sta facendo molto simile. Il punto di partenza comune è la grafica, visto che il gioco è già bello e strutturato, stabilito che la qualità artistica di questa dipende in buona parte dall'abili-

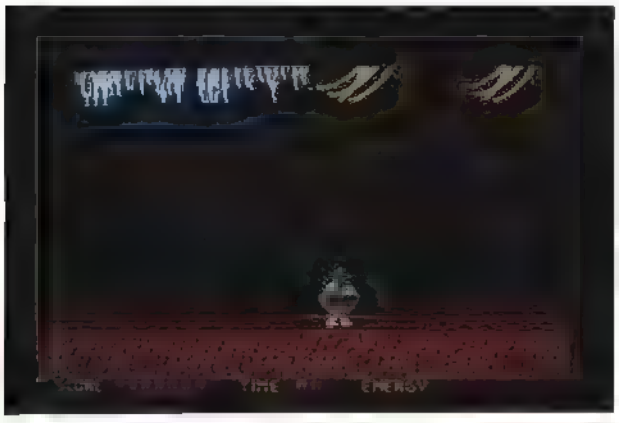
È disponibile, presso la redazione, il disco con il programma presentato in questa rubrica. Le istruzioni per l'acquisto e l'elenco degli altri programmi disponibili sono a pag. 263.

tà dei disegnatori, ci occuperemo della filosofia di gestione dal punto di vista software. Il C-64, come abbiamo più volte ripetuto, possiede un sistema di scrolling parte hardware e parte software ■ non ci soffermeremo a discuterne. L'Amiga ne possiede uno totalmente hardware, ma che necessita per forza di cose di una parte software. Tra l'altro, inizialmente nessuno si è mai preoccupato di utilizzare questa risorsa a fondo ■ infatti ■ vedeva. Il sistema utilizzato nel nostro pseudo-editor di videogiochi è in buona parte software per mia scelta (si poteva comunque fare di meglio). Esso ■ basa su un concetto abbastanza semplice, quello del fondale composto da «mattonelle» ovvero da una combinazione opportuna di rettangolini disegnati in modo da essere «componibili». Ciò per economizzare la RAM a disposizione, anche perché se è vero che l'Amiga ha 512k ■ anche vero che la grafica bitmap se la succhia come se niente fosse. A proposito l'Amiga lavora sempre come il rispettivo modo Hires del C-64; se su quest'ultimo, detto modo occupa 10k, sull'Amiga, in 16 colori, ne usa 32k e quando occorre gestire 10 schermi del genere...

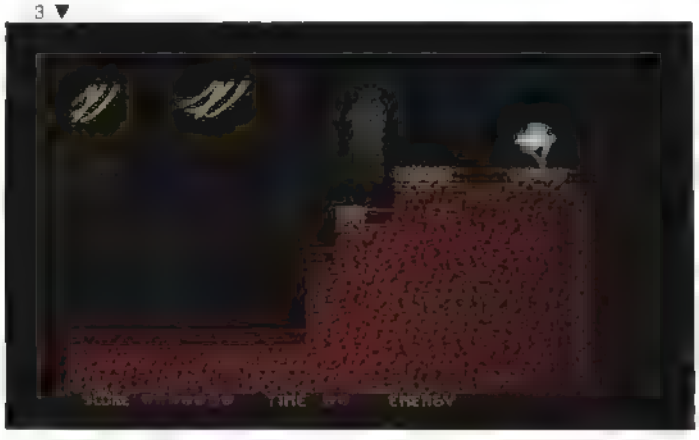
Il fondale del nostro gioco è quindi composto da tante mattonelle una accanto all'altra che «scrollano» grazie al blitter, che ad ogni passata sposta di due pixel a sinistra (se lo scrolling è richiesto) tutto lo schermo e quando i pixel diventano 16 (ampiezza di una mattonella) aggiunge ■ mattonelle in verticale a destra dello schermo (che sono ampie 24 pixel l'una per un totale di 196 pixel... restano ■ pixel verticali per scritte varie). In realtà il sistema di scrolling è un tantino più complicato ed ora gli daremo uno sguardo più approfondito. Per evitare sfarfallamenti dell'immagine il gioco utilizza due pagine di 320×200 pixel 16 colori l'una, che vengono scambiate visivamente ogni volta che quella nascosta è stata ridisegnata completamente. Esistono inoltre altre due pagine che vengono ridisegnate più lentamente ■ che effettuano un lavoro invisibile, ma estremamente utile. L'area di gioco effettivamente riempita dal fondale ■ di 304×192 pixel. Il blitter preleva un'area di 304×192 pixel da una delle due pagine sempre nascoste. Quest'ultima pagina è completamente piena per un'area di 320×192 pixel. L'altra pagina sempre nascosta viene disegnata, sempre dal blitter, in 8 passate; il blitter quindi prende di volta in volta un pezzo dalla pagina sempre nascosta completamente disegnata ■ lo stampa nell'altra pagina, ma spostato di 16 pixel ■ sinistra e in più aggiunge per ognuna delle 8 passate una mattonella a destra (in colonna). Il risultato di ciò è che al termine delle 8 passate ■ pagina che stiamo riempiendo con questo sistema ■ completamente riempita per un'area di 320×192 pixel e contiene un

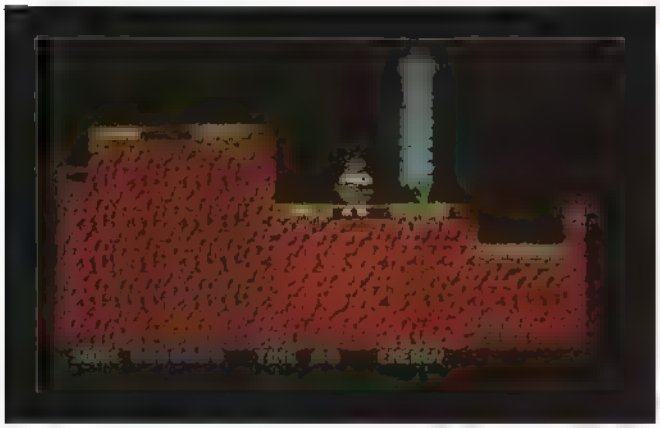
1 ▶

2 ▼

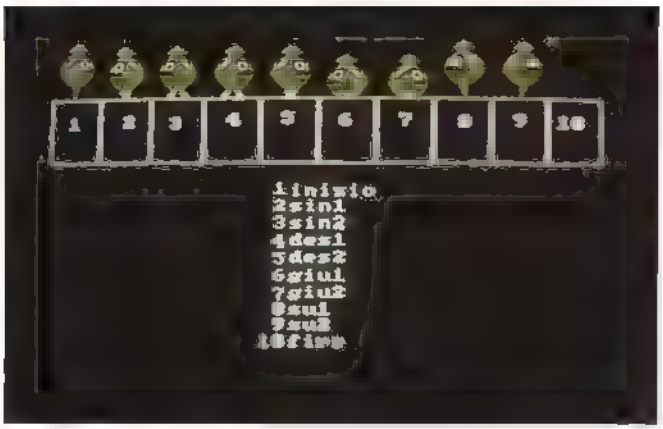


3 ▼





4



5

disegno simile ■ quello della pagina di partenza, ma spostato di 16 pixel ■ i 16 pixel che ■ sono eliminati a sinistra sono stati sostituiti a destra con una serie di 8 mattonelle.

E ancora non è finita. Nel frattempo il blitter deve, ad ognuna di queste passate, stampare un'area di 304×192 pixel sulla pagina invisibile che diventerà subito dopo visibile, di volta in volta prelevando due pixel più a destra questa fetta di grafica, dalla pagina mai visibile e completa. Potrà sembrare il sistema più complicato possibile, ma il suo scopo è quello di distribuire gli sforzi del blitter nel modo più uniforme possibile tra le 8 passate che compongono il ciclo. Oltre a questo ingrato compito il blitter è sottoposto anche alla stampa dei fotogrammi del giocatore e di quelli dei nemici.

Veniamo ■ come vengono realizzati il giocatore ■ i nemici. Sul C-64 avremmo risolto il problema dedicando uno o più sprite al giocatore e i restanti ai nemici, magari effettuando delle «moltiplicazioni». In ambiente Amiga possiamo utilizzare sprite, ma non è l'unica alternativa; è possibile utilizzare il blitter per stampare porzioni di grafica con una modalità particolare che permette di rendere immune da stampa zone particolari, che ad esempio vogliamo trasparenti; quindi in un disegno contenuto in un'area di 32×40 pixel, che rappresenta ad esempio una sfera, bisogna stabilire quali sono i pixel che non contengono il disegno e che quindi devono lasciare inalterato il fondale, al fine di evitare i tanto antiestetici «contorni neri». L'utilizzo del blitter non è dei più semplici, ma una volta realizzata una routine, anche complicata come in questo caso, basterà chiamarla ed il gioco è fatto. Ho utilizzato il blitter in quanto gli sprite sono troppo piccoli ■ hanno pochi colori. Qualcuno tra voi si sarà accorto che non ho detto come si fa ■ ripristinare il fondale una volta che uno «blit-sprite» è stato stampato... non occorre ripristinarlo in quanto con la tecnica di scrolling che vi ho descritto sopra, il fondale viene di volta in volta ridisegnato da capo, cancellando eventuali «blit-sprite» indesiderati.

Il numero complessivo di personaggi che possono essere contemporaneamente sullo schermo (nel nostro caso) è 3, più uno blit-sprite per il «proiettile» lanciato dal giocatore. Visto che uno dei tre personaggi è il giocatore, gli altri due sono i cattivoni.

Le routine di movimento del giocatore e dei nemici sono uguali sia nel caso del C-64 che in quello dell'Amiga. Anco■ diverse sono invece le routine sonore, ma solo per il modo di gestire l'audio.

Visto che l'Amiga funziona più o meno come un registratore a 4 piste basterebbe campionare una musichetta e qualche effetto sonoro per risolvere il problema... in modo discreto. Ed è proprio così che ho fatto nel mio caso, ovvero ho messo una musichetta campionata in loop come sottofondo musicale (il sistema, lo riconosco, è tra i più pigri possibili) e ho usato due effetti campionati per il giocatore (quando spara ■ quando viene colpito). Con il C-64 dovevamo realizzare una routine di gestione del chip SID o utilizzare una già preconfezionata (ad esempio l'editor musicale del sottoscritto). Il gioco è sostanzialmente tutto qui; 5 livelli diversi nella grafica ■ nei nemici una schermata grafica gradevole come premio finale e buon divertimento al giocatore.

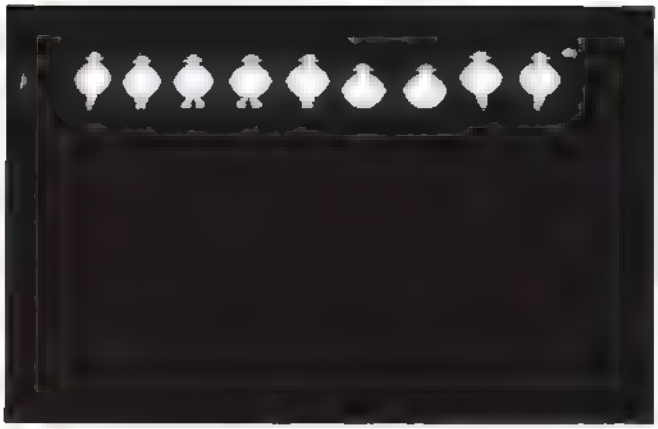
Una mano al programmatore

Ed ora è il momento di dare istruzioni a coloro che vorranno cimentarsi nella realizzazione del videogioco personalizzato con l'ausilio del mio pseudo-editor. Innanzi tutto si chiama «pseudo-editor» in quanto ■ costituito sostanzialmente da una serie di utility e non da un vero ■ proprio programma stile S.E.U.C.K. Quello che l'utente «pseudo-programmatore» dovrà realizzare è il seguente materiale:

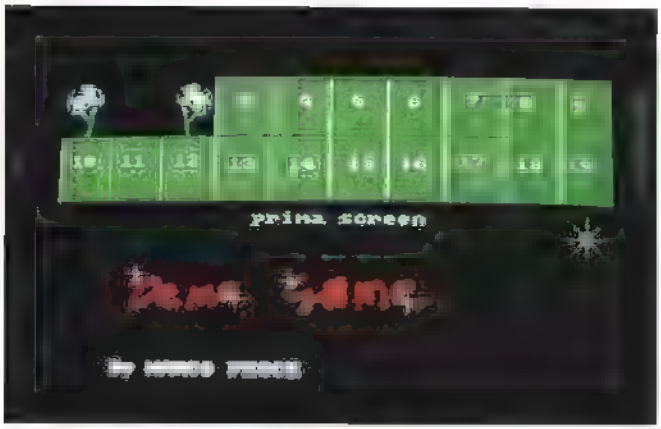
— 5 schermate contenenti le mattonelle;
— 1 schermata con i movimenti del giocatore;
— 10 schermate con quelli dei nemici;
— 5 blocchi dati per ■ distribuzione delle mattonelle;
— 5 blocchi dati per gli schemi di attacco dei nemici;
— una musica campionata;
— due effetti sonori;
— una schermata finale.

Ognuno di questi prodotti deve essere realizzato in un certo modo ■ con un determinato programma. Un solo avvertimento: l'impresa richiede un po' di abilità e quindi non è proprio alla portata di tutti, ma seguite attentamente le mie istruzioni ■ giudicate da soli se la cosa fa per voi. Magari se il vostro elaborato lo ritenete di qualità, potete inviarmelo in redazione: chissà che non... venga ricompensato.

Tutta la grafica va disegnata con il Deluxe Paint e dovrà rispettare le dimensioni che vi dirò tra un po'. Ogni schermata dovrà subire un trattamento particolare con apposite utility, ovviamente incluse nel pacchetto. Le cinque schermate relative alle mattonelle devono essere del tipo di quella in foto 8, ovvero ogni singola mattonella va inserita in una casella di 16×24 pixel, facendo attenzione a non creare «disallineamenti». Ogni schermata è valida per un solo livello. Potete, volendo, cambiare anche le scritte che vedete sotto (score, time, energy). Non variate troppo lo schermo colore delle schermate demo (pena problemi con le scritte). Realizzata la prima schermata con le mattonelle, salvatela e «convertitela» con il pro-



6



7

gramma «Convertitore» che trovate sul dischetto. Salvatela con il nome «01pic». Le successive schermate dei... successivi livelli devono essere chiamate «11pic», «21pic», «31pic», e «41pic». La «51pic» è lo screen finale.

Ed ora un elenco dei nomi da dare ai file:

musica: file della musica campionata
suonoaio: file del suono giocatore ferito
suonotio: file del suono giocatore che spara
02pic: schermata con movimenti del giocatore (vedi foto 4)
01mas: schermata con ombre del giocatore (foto 5).

File che variano con il livello:

01pic: schermata con mattonella (primo livello).
03pic: scr. sprite dei nemici1 (primo livello)
04pic: scr. sprite dei nemici2 (primo livello)
02mas: scr. ombre dei nemici1 (primo livello)
03mas: scr. ombre dei nemici2 (primo livello)
0data: dati delle mattonelle
0nem: dati dei nemici

per i successivi livelli basta sostituire la prima cifra del nome con i numeri 1, 2, 3 e 4;

51pic: schermata finale.

Tutte le schermate sono in 16 colori. La schermata con i movimenti del giocatore (che è rappresentato da un'area grafica di 32×40 pixel) deve contenere i movimenti come in foto 4, ovviamente cambiando a piacere il disegno. Anche detta schermata va convertita con il programma «Convertitore». L'ultimo fotogramma è il proiettile che spara il giocatore (a gittata limitata). Per realizzare lo screen con le ombre (foto 5) occorre catturare (siamo in ambiente DPaint) con un brush tutti i fotogrammi e renderli del colore grigio scuro (l'ultimo) con l'opzione «Color» quindi stamparli nella stessa posizione e salvare lo screen, per poi convertirlo con il programma «maschera» e salvarlo con il nome «01mas». Anche gli screen dei nemici vanno realizzati con la stessa modalità e con le ombre, solo che i fotogrammi di animazione sono solo 2 (più che sufficienti nel nostro caso); uno è quello della schermata di foto 6 e l'altro quello della schermata di foto 7 (si alternano all'infinito con una certa frequenza). Le loro dimensioni sono le stesse del giocatore.

Per realizzare le mappe (0data e 0nem) occorre il programma «Edtabelle». Prima di addentrarci nella spiegazione del suo funzionamento voglio dirvi quali sono gli algoritmi di movimento dei nemici. Essi non sparano mai, ma possono:
— cadere dall'alto e scoppiare;
— cadere dall'alto e rimbalzare;
— camminare a destra (evitando ostacoli);
— camminare a sinistra (evitando ostacoli);
— restare fermi.

Solo uno a scelta di questi algoritmi può essere attribuito a ciascun nemico. I nemici entrano in azione in funzione della posizione dello scrolling ed è in base ad essa che verranno programmati.

Quando un nemico viene colpito dal proiettile del giocatore esso esplode con il fotogramma a forma di stella delle foto 6 e 7.

Il giocatore muore quando finisce l'energia o il tempo.

E veniamo all'editor di tabelle «Edtabelle». Innanzi tutto va caricato con l'AmigaBasic. Subito dopo il caricamento ci viene chiesto il nome della schermata che contiene le mattonelle (nel caso del primo «quadro» il nome è «01pic») quindi il nome della tabella-mattonelle (se per caso ne avevamo già iniziata una) e quello della tabella-nemici. Cominciamo con la realizzazione della tabella-mattonelle. A questo punto occorre improvvisarsi equilibristi (sto scherzando) per riuscire a capirci qualcosa tra tutte le schermate che possono visualizzarsi. Innanzi tutto occorre rendere visibili i gadget del Workbench che permettono lo scambio tra gli schermi (quelli in alto a destra) e quindi scambiare il Workbench con lo screen sottostante. In questo dovremo comporre il nostro «mosaico», servendoci della schermata sottostante, contenente il disegno delle mattonelle. Per passare da uno screen ad un altro è sempre necessario servirsi dei gadget di Intuition quindi fate attenzione perché negli screen sottostanti il Workbench questi non sono visibili (occorre andare a memoria). Ma vediamo come si compone il mosaico. Il puntatore del mouse controlla una mattonella base che può essere depositata sul terreno semplicemente premendo il tasto «P». Per cambiarla basta andare nello screen delle mattonelle e premere il tasto «M» su quella desiderata (per poi tornare al «foglio da disegno»). Ovviamente è anche possibile scegliere di «tirare giù» lo schermo «a tendina», per vedere sia le mattonelle che il foglio da disegno (vedi foto 9). La metà a destra dello schermo è quella effettivamente disegnabile, mentre quella a sinistra serve per avere un sistema di riferimento. Premendo la barra spazio si memorizza la mezza schermata a destra e verrà effettuato uno scrolling grosso di mezza pagina. A questo punto potete continuare a disegnare. È anche possibile effettuare uno scrollig manuale per le correzioni o per visualizzare una particolare zona della mappa-tabella; basta servirsi dei tasti del tastierino numerico «4» per sinistra e «6» per destra con salti di mezzo schermo. C'è anche una funzione di

8

9

GOTO alla pagina numero tal-dei-tali; si preme «G» e si inserisce il numero desiderato. Ci sono notevoli possibilità di rendere più rapida la composizione della mappa, prelevando fette più grandi di una mattonella (gruppi di mattonelle) sia dallo screen che le contiene che dalla mappa stessa (una specie di funzione brush del DPaint). Per fare ciò è necessario servirsi dei tasti:
«1» - per indicare la mattonella di start;
«2» - per indicare quella di end (prelevando dalla mappa);
«3» - per indicare quella di end (prelevando dallo screen mattonelle).

Per stampare questo gruppo di mattonelle basta premere il tasto «I» (dopo aver ovviamente «mirato» nel punto desiderato). Con il tasto «0» si preleva la mattonella base direttamente dallo screen della mappa.

In qualunque momento è possibile salvare («SAVE») la tabella premendo «shift»+«S» e quindi rispondendo alla richiesta del nome.

Terminata la composizione (circa 15 schermate di ampiezza orizzontale), occorre definire quali tra le mattonelle dovranno risultare «solide», ovvero impenetrabili, sia da parte del giocatore che da quella dei nemici. È chiaro che già in fase di disegno si deve studiare più o meno quali saranno dette mattonelle. Esse impediscono di essere attraversate e quindi ci si può camminare sopra (con gli sprite!) e possono limitare l'avanzata e costringere a «salti». È importante non disporre più di 3 mattonelle solide in verticale, pena un'impossibilità di salto da parte degli sprite. Vediamo come si definiscono; bisogna entrare in questa modalità premendo «shift»+«M», quindi occorre andare sullo screen delle mattonelle e puntarle con il mouse, premendo il tasto «M» per confermare. Terminata la scelta si preme lo spazio e inizia il processo di «solidificazione», che durerà un bel po' (circa 15 secondi per mattonella solida). Ad ogni mattonella completata si udirà un «beep».

Terminata questa operazione potete salvare finalmente il blocco definitivo della tabella-mappa. Se per caso avete solidificato mattonelle sbagliate, basterà entrare nel modo di disattivazione della solidificazione, premendo «shift»+«N» e eseguendo lo stesso tipo di selezione. Quando una mattonella viene definita come solida essa si deforma per indicare che viene definita come solida essa si deforma.

Passiamo ora all'edit del movimento dei nemici. Per entrare in questa modalità occorre premere «shift»+«Q». A questo punto fate «scrollare» lo schermo di una mattonella alla volta con il tasto «A» fin quando non arrivate alla posizione da voi scelta per far apparire il primo nemico; premete il tasto «I» e il programma vi chiederà quale deve essere il tipo di movimento (da 0 a 4). I numeri corrispondono a:

0 - nemico che cade e si «rompe»
1 - nemico che cade e rimbalza
2 - nemico che avanza verso sinistra
3 - nemico che avanza verso destra
4 - nemico che resta fermo nella posizione di partenza.

Scelto il tipo di movimento il programma ci chiede di posizionare il giocatore sullo schermo, quindi dobbiamo tornare alla mappa e puntare il mouse, tenendo conto che la megamattonella che state spostando è delle dimensioni di un nemico.

Segue la richiesta che stabilisce quale sarà lo sprite di animazione (e dovete farvi un elenco di dove sono posizionati nelle vostre schermate; i numeri si contano da sinistra a destra, come in foto 8 e 7). Siamo giunti alla conclusione del blocco dati per il primo nemico; ora basta ripetere l'operazione per tante volte durante tutto il percorso-mappa. Tenete comunque presente che se sullo schermo dovrebbero esserci più di 2 nemici, il primo di questi che è stato attivato verrà fatto «esplodere» (quindi non esagerate con gli attacchi). Per salvare questa tabella basta premere «shift»+«S» e impostare il nome. La collisione giocatore-nemico comporta un decremento continuo di energia; la collisione proiettile-nemico comporta un incremento del punteggio.

Mancano questo punto solo informazioni su i file sonori.

La musica campionata deve avere una lunghezza di 26690 campioni e su sampling rate di 8363. Il «suonoaio» è lungo 3778 e ha lo stesso sampling rate. Il «suonotie» è lungo 1794 e ha un sampling rate di 9800.

Sul dischetto entrano a mala pena tutti i dati delle schermate e delle tabelle, quindi bisogna farne una copia e ripulirla delle utility. La stratup-sequence è già predisposta per far avviare il gioco in autostart.

Per usare i programmi-utility occorre caricare il Workbech e fermarlo con «CTRL»+«D» per poi inserire il dischetto contenente i programmi e avviarli da CLI. Tutte le schermate e le «ombre» devono essere salvate sullo stesso dischetto contenente le utility e poi copiate su quello del gioco, pena una parziale ma dannosa cancellatura delle stesse. Vi auguro di cuore buon divertimento.

di Marco Pesce

Un videogioco tutto nostro

È finito! Potrei anche concludere così questo ultimo appuntamento con l'interminabile serie di articoli sul Megagame 64, ma sarebbe una mossa scorretta nei confronti di tutti quei lettori che vogliono il gran finale. Scegliete voi cosa leggere per primo; l'elenco dei partecipanti, la cronologia, oppure le istruzioni per l'uso? Per il lettore che non ha mai comprato MC in questi ultimi due anni (abbondanti), la cosa migliore da farsi sarebbe quella di cominciare con una breve introduzione. Calcolando che tutti gli altri comincerebbero a fischiare consiglio ai primi di dare uno sguardo alla parte cronologica. Noi invece cominciamo con le «istruzioni»

Che gioco è?

Il Megagame 64 è un videogioco che richiede obbligatoriamente la «presenza» di due giocatori umani. Ciascuno è alla guida di un complesso di 8 pianeti. Lo scopo è quello di annientare l'avversario, conquistando i 16 pianeti del sistema. Non ci sono sezioni arcade, quindi il gioco è tutto strategia.

Come si gioca?

Occorrono due joystick da inserire nelle rispettive porte. Ciascun giocatore controlla un cursore che può essere posizionato liberamente nella propria porzione di schermo. Tramite un sistema di pannelli possono essere gestiti i seguenti comandi base:
— viaggi tra i pianeti
— costruzioni sui pianeti
— guerre tra i pianeti.

Entrambi i giocatori hanno le stesse possibilità.

Una mappa, visualizzata alla destra dei due subschermi, ci indica la posizione, lo stato (guerra/pace/allarme a seconda del segnalino, rispettivamente un asterisco, un punto, un triangolo), il possessore di ciascun pianeta. Posizionando il cursore su uno dei pianteti e premendo il tasto fire ci vengono elencate una serie di caratteristiche del pianeta stesso:
AL: alimenti presenti sul pianeta
EN: energia
MI: minerali
AB: abitanti
AR: armi
P : potenza produttiva del pianeta
SU: superficie utilizzata
ST: superficie totale.

Se il pianeta è in stato di allarme vuol dire che qualcosa non va nel suo ciclo produttivo (terminata una risorsa, fine dello spazio disponibile, ecc).

Il ciclo produttivo di un pianeta

Lo scopo fondamentale è quello di ottimizzare la produzione di armi e... abitanti. Questo perché per poter attaccare un pianeta occorrono appunto queste due «risorse», in quantità più grande possibile. La risorsa indispensabile alla vita dei pianeti è l'energia, grazie alla quale vengono rifornite le industrie e vengono effettuati i viaggi e le costruzioni. Grazie all'energia vengono prodotti minerali, che unitamente alla prima, riforniranno la produzione di armi e alimenti, questi ultimi indispensabili per la riproduzione degli abitanti. Su ogni pianeta quindi c'è una certa quantità di installazioni produttive, che può essere modificata dal giocatore. Il potere produttivo del pianeta rappresenta la potenza che hanno su quel particolare pianeta dette installazioni (invariabile nel tempo).

Costruire strutture

Questa operazione si può effettuare premendo il tasto «S», visibile sopra la mappa vicino agli altri due tasti (viaggi e guerre). Apparirà quindi, sulla sinistra, ciò che potete ammirare nella foto 2 (in alto). Accanto alle sigle (AL, EN, ecc.) appariranno delle barre (come per la descrizione del pianeta) che indicheranno la quantità di installazioni presenti sul pianeta (nel caso di foto 2, nessuna!). La sigla «s» sta per spazio liberato o occupato, a seconda che l'operazione sia una distruzione o una costruzione. La sigla «c» indica il costo (in energia e

È disponibile, presso la redazione, il disco con il programma presentato in questa rubrica. Le istruzioni per l'acquisto e l'elenco degli altri programmi disponibili sono a pag. 279.

minerali) della operazione. I tasti-frecce servono per variare le quantità, mentre il tasto «R» serve per ripristinare lo stato iniziale del pianeta (reset), ovvero per annullare le modifiche che abbiamo fatto (magari involontariamente) alle strutture. Con il tasto «C» si conferma la modifica.

Effettuare viaggi

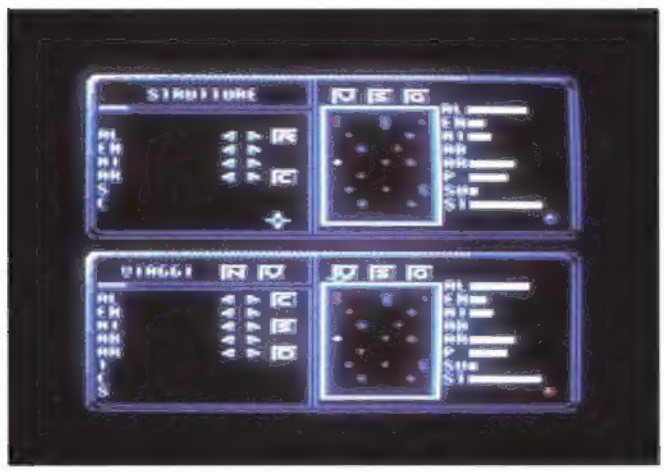
Nel caso uno dei vostri pianeti sia rimasto senza una particolare risorsa (soprattutto armi/abitanti o energia) è possibile rifornirlo tramite un altro pianeta, che magari ha il problema opposto, con un viaggio di materiali. Il pannello è quello che si attiva con il tasto «V» della sezione mappa (foto 3).

Come potete osservare dalla foto, ci sono due tasti accanto alla scritta «VIAGGI», che non erano presenti nell'opzione «STRUTTURE» e che ci permettono di passare alla gestione di due subopzioni: «viaggi Nuovi» e «viaggi Vecchi». Nel caso vogliamo occuparci di un nuovo viaggio basta premere la «N» o agire sulle quantità di materiali e stabilire il pianeta sorgente e quello destinazione (tasti «S» e «D»), quindi confermate (tasto «C»). Vengono indicati il tempo necessario (a seconda della distanza tra i due pianeti), il costo (in energia) o lo spazio liberato sul pianeta sorgente, quindi occupato su quello destinazione. Quando si preme il tasto «S» (o il tasto «D») esso si illuminerà e sulla mappa apparirà il pianeta che abbiamo scelto, quindi possiamo cambiarlo. Passando al pannello dei «viaggi Vecchi» possiamo osservare l'elenco dei viaggi che abbiamo avviato e il tempo che manca alla loro conclusione.

Dichiarazione di guerra

Nella foto 4 potete osservare i due pannelli che riguardano le guerre «nuove» (in alto) e quelle «vecchie» (in basso). Come abbiamo detto ci occorrono sia armi che abitanti, e la «P» che vedete nel pannello simboleggia questa quantità (potenza). Il pianeta sorgente e quello destinazione si selezionano con lo stesso sistema dei viaggi; del resto si tratta effettivamente di un viaggio, che quindi richiederà anche una porzione di tempo e una quantità di energia.

Foto 2 ▶



◀ *Foto 3*

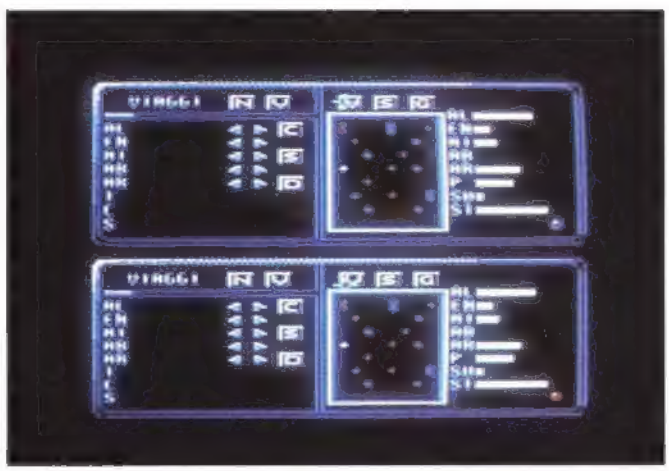


Foto 4 ▶

Gli sconti

Il pannello delle guerre «vecchie» visualizza la potenza del pianeta sorgente (attaccante) e quella del destinazione (attaccato), quindi il tempo che manca all'arrivo, ovvero all'inizio dello scontro. Per visualizzare una particolare guerra basta selezionare tramite mappa il pianeta che ci interessa (in guerra) e poi, con le frecce spostarsi per tutto l'elenco delle spedizioni fatte contro tale pianeta. Il più forte quasi sempre vince, ma non è detto, perché gli scontri avvengono con una percentuale di probabilità (ad armi pari equivalente al 50%) e potrebbe accadere l'imprevedibile.

Il tempo

Il tempo scorre secondo un particolare sistema di riferimento; ogni tot cicli lo schermo visualizza una schermata simbolica che sta a rappresentare la fine del «giorno». Dopo qualche secondo il gioco ricomincia e i pianeti avranno subito una variazione nella loro struttura. Come vi sarete resi conto ci sono un bel po' di cosette da fare e quindi il divertimento non dovrebbe mancare. Siamo giunti alle inevitabili conclusioni. Lo scopo dell'operazione Megagame 64 è stato raggiunto in maniera soddisfacente (per non sbilanciarsi) e da parte mia mi ritengo più che soddisfatto. Possiamo quindi cominciare un nuovo megagame... stavo scherzando!!! L'unico punto negativo potrebbe essere proprio il tempo che abbiamo impiegato a realizzarlo, ma in fondo era un esperimento, quindi non lamentiamoci troppo.

Cronologia del Megagame 64

Dicembre '87 (69): comincia la «nostra avventura nel magico mondo dei videogiochi. Vengono invitati i lettori ad inviare un'idea per un videogame «fuori dal comune», che dovrà essere realizzato con la loro collaborazione.
Gennaio '88 (70): pausa in attesa della corrispondenza dei lettori.
Febbraio '88 (71): arrivano le prime proposte ma...
Marzo '88 (72): ...è con il numero di marzo che esplode l'inventiva dei lettori. Personalmente scelgo il gioco di Sergio di Fusco.
Aprile '88 (73): si comincia con una impostazione generale del gioco.
Maggio '88 (74): fermi tutti! Questo gioco si deve cambiare. E infatti si cambia; primo schema rudimentale dell'attuale Megagame.
Giugno '88 (75): secondo sviluppo con presentazione di alcune screen grafiche, riguardanti il gioco.
Luglio/agosto '88 (76): per inserire un po' di arcade in un gioco tutto strategia viene proposta un'aggiunta: l'azione sarà realizzata con sottogiochi stile «RASTAN».
Settembre '88 (77): si discutono i suggerimenti di Marco Marinai, un lettore che propone (tra l'altro) di imitare lo stile del videogioco «SHADOWFIRE».
Ottobre '88 (78): pausa.
Novembre '88 (79): si opta per un gioco con grafica ridotta all'osso ■ si abbozzano i primi pannelli.
Dicembre '88 (80): nella speranza (vana) di non arrivare al «prossimo dicembre» c'è il successivo sviluppo dei pannelli.
Gennaio '89 (81): si cambia grafica ■ si presenta quello che in seguito diventerà lo stile grafico definitivo.
Febbraio '89 (82): si modifica un po' il gioco e radicalmente lo stile grafico. Si aggiunge un sottogioco tipo «SPACE HARRIER».
Marzo '89 (83): provo a lanciare una trovata che dovrebbe rivitalizzare il colloquio con i lettori; si tratta di un collegamento telefonico... (poco dopo disattivato).
Aprile '89 (84): la grafica viene momentaneamente ridimensionata e così è anche per la struttura dei pannelli. Niente giochi arcade.
Maggio '89 (85): primo riepilogo dello «stato dei fatti» e tentativo di stringere i tempi. Invito ai lettori per la scrittura di una storia fantascientifica da abbinare al videogioco.
Giugno '89 (86): presentazione di schermi grafici per le animazioni da inserire tra un pannello e un altro.
Luglio/agosto '89 (87): pausa...
Settembre '89 (88): ... che durerà ...
Ottobre '89 (89): ... fino a ottobre.
Novembre '89 (90): viene presentata la struttura definitiva del Megagame 64 e la grafica (schematizzata) dei nuovi pannelli. Alla programmazione dovrà pensare il solo Marco Pesce.
Dicembre '89 (91): le prime routine della presentazione con grafica praticamente definitiva.
Gennaio '90 (92): pausa.
Febbraio '90 (93): i pannelli con lo stile grafico del numero di gennaio '89.
Marzo '90 (94): pausa
Aprile '90 (95): il Megagame è finito!

Chi ha partecipato

Quello che segue è un elenco (in ordine cronologico) di tutti coloro i quali hanno in qualche modo dato un contributo, sia pur minimo, alla realizzazione del Megagame 64. Speriamo ci siano tutti:
Paolo Costabel
Fabrizio Baldini
Stefano Innocenti
Fulvio, Flavio e Stefano Chini
Gianni Zambon
Daniele Argento
Umberto Michelucci
Franco Violante
ITALIAN TEAM
Franco Orsogna
Jacopo Piazzi
Sergio di Fusco
Alberto Blanchi
Luca Persico
Vito Armenise
Alberto Barbero
Emilio Orione
Gianfranco Gramiglia
Gianni Manenti
Enrico Ferrante
Roberto Morassi
Tommaso e Luigi Bini
Michele Signorile
Marco Marinai
Matteo Bittanti
Stefano Degan
Nicola Colella
Claudio Parmigiani
Emiliano D'Onofrio
Ciro Cane'
Bruno Saetta
Conan of CCS
Giulio Ceriola
Ivan Melle
THE BEST THUNDER
Giulio Cellitti
Andrea Beltrame
Massimiliano Pigozzi
Pierpaolo Bergamo.